Spring 7-1-2008

Measuring and Visualizing Anonymous Network

Jie Yu
Dakota State University

MEASURING AND VISUALIZING ANONYMOUS NETWORK

A graduate project submitted to Dakota State University in partial fulfillment of the requirements for the degree of

Master of Science

in

Information Systems

July, 2008

By

Jie Yu

Project Committee:

Dr. Ronghua Shan

Dr. Steve Graham

Dr. Xinwen Fu

PROJECT APPROVAL FORM

We certify that we have read this project and that, in our opinion, it is satisfactory in scope and quality as a project for the degree of Master of Science in Information Systems.

Student Name: Jie Yu

Master's Project Title: Measuring and Visualizing Anonymous Network

Faculty supervisor: Ronghua Shan Date: 8/24/2009

Committee member: Date: 8/24/2009

Committee member: Date:

ABSTRACT

My project is about anonymous communication network. I took one of the most popular anonymous communication tools, Tor as the example to analyze and test the existing Tor anonymous network, and then found out the weakness of Tor network. This analysis can help Tor program developers improve the performance of Tor network.

In this thesis, first I will represent the structure and principle of Tor network, and then introduce the Tor network software and its installation.

After analyzing the existing Tor network, I set up a database to store comprehensive information about all Tor server nodes. The information includes IP address, Nick Name, Publish time, Publish date, all flag statuses, and so on. I also wrote several PHP scripts to update the database automatically according to the scheduled time with Crontab application under Linux system.

At the same time, I set up a website using HTML, CSS, and PHP for presenting the nodes' information and for website visitors to retrieve some nodes with some conditions. Visitors can select one of many search keywords or write their own SQL statement to retrieve the nodes that satisfy their desire.

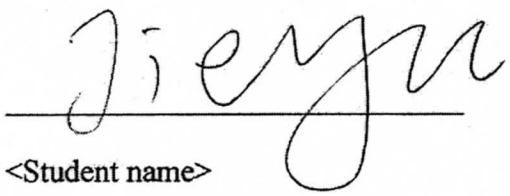
My project is based on LAMP technique combination that are Linux (Fedora), Apache, MySQL, and PHP. I combined all these four tools tightly to visualize the information of Tor network and try to improve the use of Tor anonymous network.

DECLARATION

I hereby certify that this project constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions or writings of another.

I declare that the project describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,



<Student name>

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Background of the Problem

With the rapid increase of the use of the Internet, people consider more about their important and private information that are transmitted on the Internet when people use the Internet for work and daily life activities such as shopping online, managing bank accounts online, visiting private website without being known by others, chatting privately with somebody using instant messaging application, and so on.

My project is about the anonymous communication network. Anonymous communication has been used on the Internet for several years. There are a lot of advantages using it, such as improving Internet users' privacy and security, keeping websites from recording and tracking visitors' behavior, and breaking the censor block from the ISP.

With above advantages, there should be a mass of users using this excellent Internet tool. But the fact is that the users of anonymous communication tool are very few compared to the huge quantity of the Internet users because of the following two reasons.

First, it is very hard to understand the principle and structure of anonymous communication for existing and potential users because there are very little existing material and information about it. Second, the performance of anonymous communication tool is not so good that users are pleased to use it. The speed of Internet browser through anonymous communication tool is slow compared to the one without it. These two factors impact the wide use of anonymous network.

Statement of the problem

Through researching the principle and structure of the anonymous communication network, we know that anonymous communication network is a very secure and stable communication tool. But why is the Internet visiting speed through it very slow?

In my project, I tested and analyzed one of the most popular anonymous network tools, Tor. I set up an experiment system to test the existing Tor anonymous network and present the reason why Tor suffers low network throughput performance. After finding out the reason of its low performance, I visualized Tor anonymous communication network in order to promote the use of it. I set up a database providing comprehensive information of Tor network and published a website which links to this Tor network database and retrieves the information of Tor network nodes. The website and database can help Tor users get to know the principle, structure, and information about Tor network and improve the use situation of Tor network.

Objectives of the project

In my project, I found out the reason of Tor network's low performance under my advisor's instruction. After that I set up a website and a database about the information of Tor network nodes to promote the use of Tor network. I presented the reason in my project to help Tor software programmers change the software codes to improve the performance of Tor network. Also I did some web programming work with Linux, HTML, PHP, and Apache environment to set up the information website of Tor network nodes. In the background of this information website, I used PHP and MySQL to make a Tor server node database that includes comprehensive information about Tor network nodes. The website and database let

visitors visualize the current status and information of Tor network and let them retrieve the specific information of one or several Tor network nodes. The objectives of finding out Tor low performance reason, information website, and database are to help improve the use of Tor network.

CHAPTER 2

LITERATURE REVIEW

Anonymous communication network review

With more and more communication on the Internet, there is much higher request for private and secure Internet communication. When we surf the Internet or chat with others using instant messaging application, we consider more private and secure problems and want to protect our privacy better.

Anonymity is righteous and necessary in many scenarios, such as protecting Internet user privacy, improving system security, bypassing Internet censorship, satisfying some antivirus requirement, and protecting Internet users' computer from hackers' attacks. With the wider use of the Internet and more improving computer hacker technologies, the Internet and network users are anxious to gain more powerful ability to keep their privacy and security. They want to request a better network tool or technology to protect their private information from being monitored by the Internet sniffers and hackers. With this requirement, anonymous communication has become more and more popular on the Internet.

As one classic anonymous communication tool, Tor gets more implementation on the world. There are many companies paying much attention to network communication security and using Tor network now. Indymedia Group recommends that their employees use Tor

when communicating with others on the Internet. Recently some departments of US Navy used Tor communication network for the activities in Middle East area. [1]

At the same time, besides the positive side of Tor communication network, there are some reports about the weakness of Tor too. For example, in September 2007, Dan Egerstad, a Swedish security specialist, found that he can capture a lot of sensitive information such as usernames and passwords for emails account though monitoring the data packets transmitted from Tor exit nodes.[2] With the design principle of Tor network, there is no encryption like SSL technology between exit nodes and the final destinations. Although this attack does not violate the anonymity for the source and destination of Tor network nodes, it is a big weakness of Tor network.

---

[1] http://www.torproject.org/overview.html.en

[2] http://en.wikipedia.org/wiki/Tor_(anonymity_network)

CHAPTER 3

SYSTEM DESIGN (RESEARCH METHODOLOGY)

Tor Network

Tor network introduction

Tor is the second generation onion router system with many advantages compared to the previous generation.

Tor is one of anonymous communication tools. It is a virtual tunnel for people on the Internet to communicate privately and securely. Over the public networks such as the Internet, Tor can keep the communication between people safe. Tor can be used with many computer applications including several web browsers, instant messaging, and so on. It can improve the privacy and security when people use these applications. There are several applications of Tor described as follows:

Tor users can surf websites without someone tracking or spying the content and information they visit. Tor user can use instant messaging application without any leak for the conversation contents and objects. Tor users also can penetrate the blocked website by Internet service provider and do not leave any identity information to that Internet service provider. When using Tor to publish a website, Tor users can get a hidden service that publishes the website but does not let visitors know the location information of this website. Tor users can avoid some harassment when they chat in some chat rooms or in some chat

forums. Journalists or staff of some organizations can use Tor to finish their work abroad without letting the foreign spies know their work contents and their employers.

"Traffic analysis" is a common surveillance on the Internet. It can disclose the address and location information of communication parties. There are two parts in one Internet data packet. One is the data payload part and the other is packet header for Internet routing. Data payload is the content that we want to transmit to the recipient. E.g. a web page content, one transmitting file, one email message, and so on. Data payload part is encrypted commonly. But the routing header of the Internet packet is not encrypted. It records the source and destination information of the packet such as the IP addresses of sender and recipient. Traffic analysis use the weakness of the header part of Internet packet to get the source and destination information of the packet. Figure 1 is the structure of IP packet.

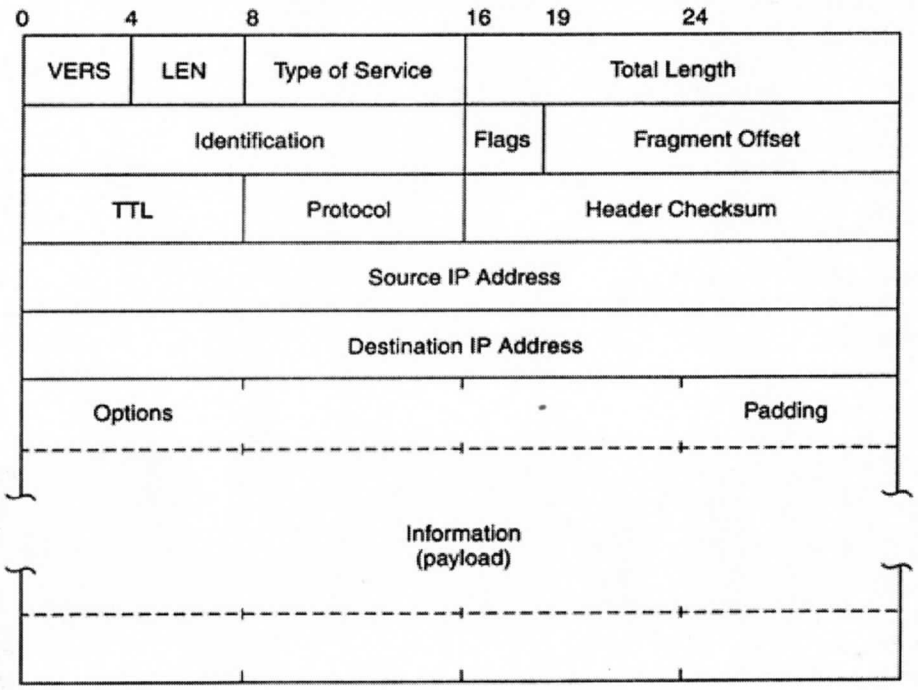


Figure 1 IP packet structure[3]

[3] http://media.techtarget.com/digitalguide/images/Misc/voip_5.gif

The header part reveals the packet's source, destination, size, timing, and so on. The recipient of the packet can know much information through checking the header. If some intermediaries sit between the source and the destination, it also can get above information easily through looking at the header.

With the disclosure of source and destination information, the Internet users may suffer some loss, for example some e-commerce website may give different prices according to buyers' IP addresses. Another instance is that eavesdropper can know your nationality through traffic analysis for IP packet header and apply some terrorism attacks aiming to some countries' users.

How does Tor anonymous communication network overcome the traffic analysis to hide the address information? Tor network distribute the Internet data packets through a sequence of nodes from the packets' source to destination. These Tor network nodes are independent and can not know the whole path of the data packets. So no one point can link the source and destination. It is just like using a twisty route to get rid of any eavesdropper who is tailing your data packets. At the same time, to confirm the privacy and security, Tor users select random nodes from Tor network system to firm the pathway from source to destination. After some time, Tor users change selected nodes to set up a new pathway to keep the pathway from monitoring. We will discuss the setup of Tor pathway as follows.

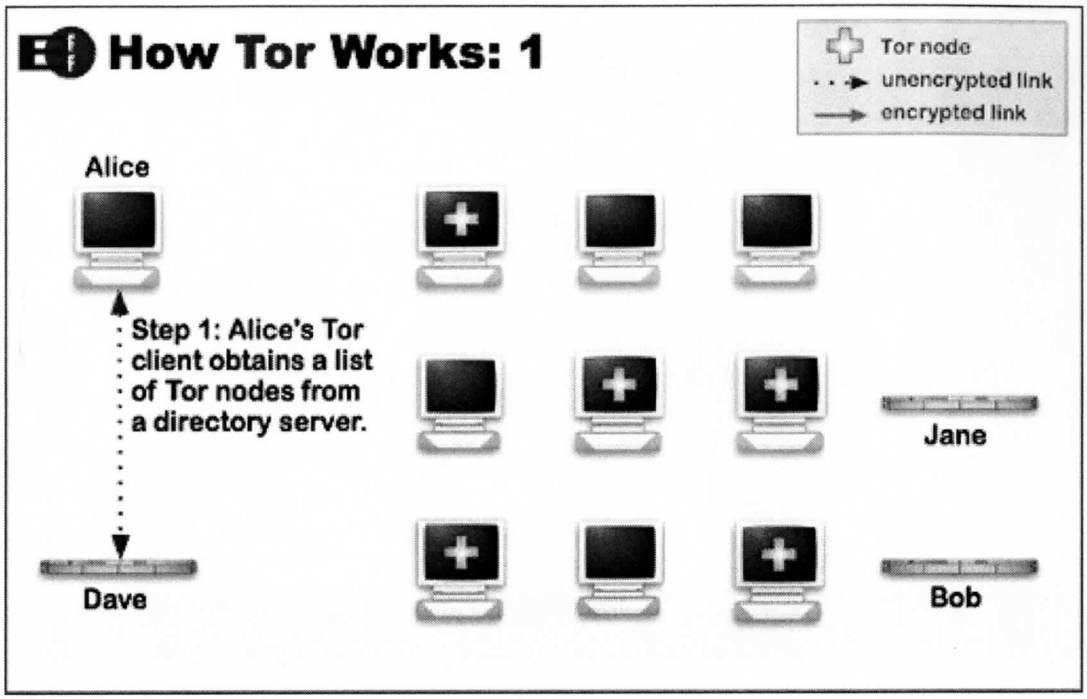


Figure 2 Tor router setup step 1 [4]

There are three kinds of Tor nodes on Tor anonymous communication network. One is server node which not only can visit their own destination websites but also can relay other Tor nodes' data packets. Server can initiate a Tor router and can be the mediate relay point for other Tor nodes' router. Another kind of Tor node is client node which only begins a Tor router and can not relay other nodes' data packets. The third one is directory server of Tor network which stores the information of all Tor server nodes. As the presentation of above figure, Tor client, Alice wants to visit the website Bob. Alice is a Tor client node only. Between Alice and Bob, there are many Tor server nodes. First, Tor client node, Alice needs to get enough quantity of Tor server nodes' information from a directory server node, Dave

[4] https://www.torproject.org/overview.html.en

here. When the client node gets enough server nodes' information, it can begin to set up Tor routers or circuits from itself to destinations.

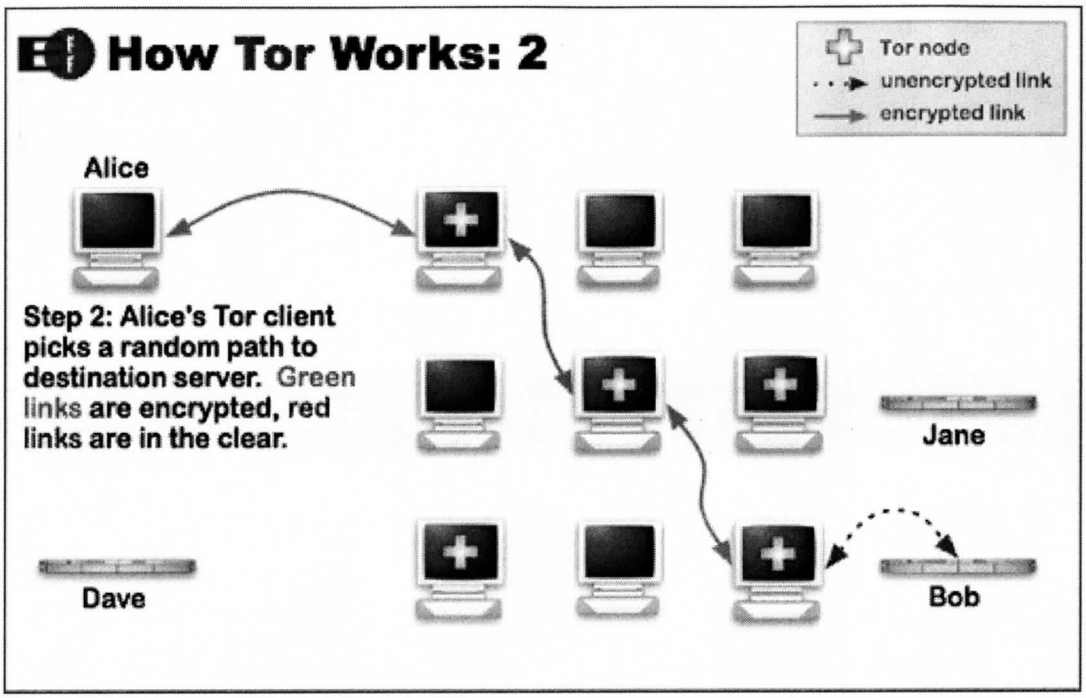


Figure 3 Tor router setup step 2[5]

After the client node gets enough quantity of server nodes, the client node builds a Tor circuit with 3 other server nodes incrementally. This circuit is set up step by step with TLS encryption and private key. The client node extends the circuit one hop at a time. Each server node along the circuit only knows the information of nodes which give date to it, and to which it gives data. No any server node knows all the nodes' information through the circuit.

After these Tor circuits are set up, the client node can transmit its data packets through them. Every circuit can transmit several TCP data streams which are Internet requests such as

[5] https://www.torproject.org/overview.html.en

http stream request and so on. This method that several streams share one circuit is very high-efficient for Tor anonymous communication network. Through these 3 mediate server nodes, the client node can exchange data with the final destination privately and securely without disclosing any sensitive address information such as IP address, packet time, and packet size. The data packets transmitted by the client node are like a three-layer onion with encryption. One hop along the circuit only can peel one layer with its private key. Every hop through the Tor circuit only knows its previous and next point. The Tor network data packet structure is as follows:

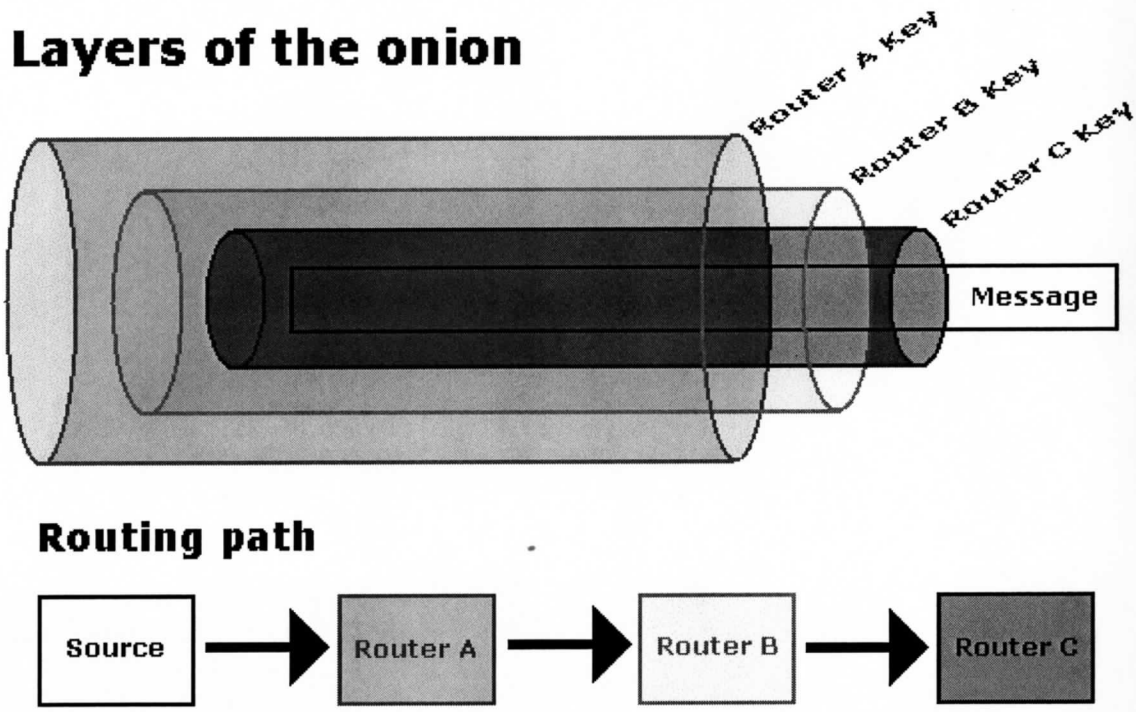


Figure 4 The structure of Tor data packet[6]

[6] http://en.wikipedia.org/wiki/Image:Onion_diagram.png

The last mediate server node of Tor circuit is called exit node. From the initial client node to the exit server node, the data packets are encrypted by TLS encryption. The data packet is not encrypted from exit node to the final destination. As we discussed before, there are some weakness for this unencrypted segment of Tor circuit. Some sniffers can collect some sensitive information of the data packet although they can not know the detailed address and location information.

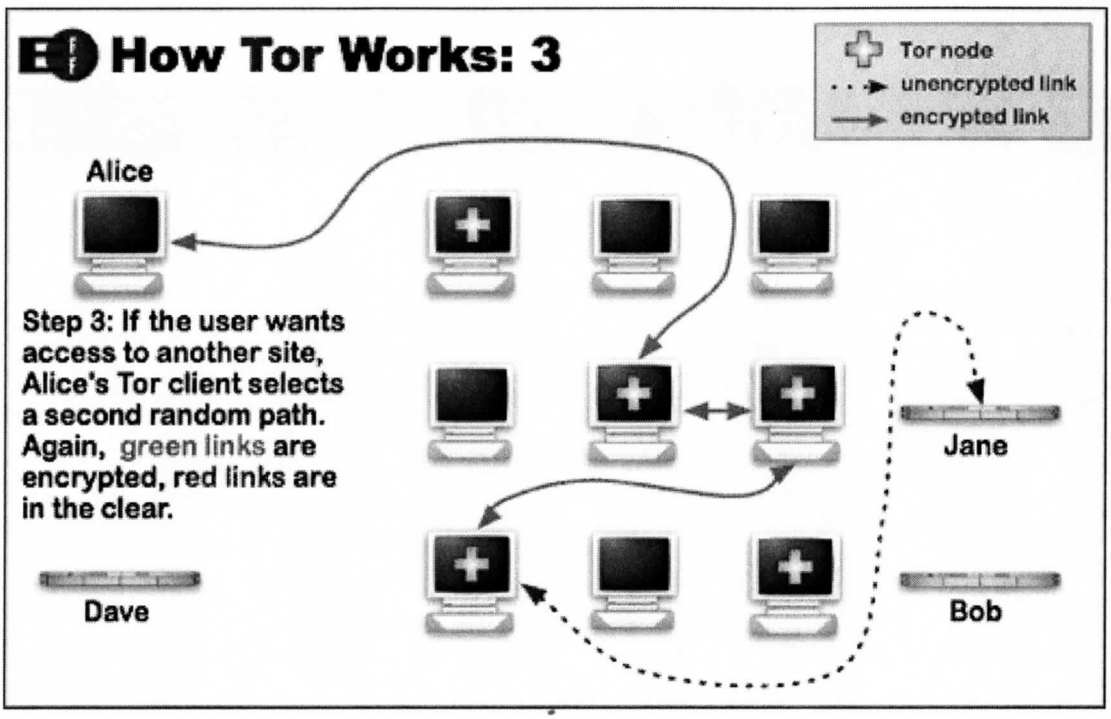


Figure 5 Tor router setup step 3 [7]

In order to be high efficient, every Tor circuit is used about ten minutes for many connections. But to keep network sniffers from monitoring Tor users according to their earlier circuit information, every ten minutes Tor client user will set up a new Tor circuit for new

[7] https://www.torproject.org/overview.html.en

TCP steams. On the above figure, when the client node Alice used one Tor circuit about ten minutes, and she wants to visit another website Jane, the client node Alice will select three different server nodes to set up a new Tor circuit to transmit data packets to web server Jane.

Tor anonymous communication network use above methodology to transmit network data among Tor nodes. Tor network is an excellent supplement tool for all kinds of encryption ways that encrypt data packet payload part. It improves the privacy of packets' location, address, time, and size information dramatically. Tor network achieves anonymity through the distributed and twisted route. Next we will discuss Tor network node types.

Tor network node classification

There are several kinds of nodes on Tor anonymous communication network. They are directory authority, directory caches, server nodes, and client nodes. We just mentioned Tor server nodes and Tor client nodes previously. These two nodes are the most common nodes on Tor network. Tor client nodes only initiate an anonymous communication whereas Tor server nodes not only begin one anonymous communication but also relay other nodes' data packets. Besides these two common Tor nodes, there are two other important nodes on Tor network, directory authority and directory cache. They take the key task of spreading information of valid Tor server nodes.

Directory authority is the information source of all valid server nodes for all other nodes on Tor network. There are several, about five, default directory authorities which are bundled with Tor installation software. When Tor software is installed in computers, these default directory authorities' information is stored in Tor application too. The information of default directory authorities includes the some Tor server nodes' information. When the

computer installed Tor application runs Tor, it will check whether there is new information about server nodes on directory authorities. If there is some new information, this computer will get the latest server nodes' information through the directory authorities.

Along with the rapid increase of Tor network, the download tasks on directory authorities become more and more heavy. There are so many Tor nodes to download Tor nodes' information through only several directory authorities. The speed of download becomes very slow. So with the development of Tor network, a new kind directory node comes out. It is the directory cache. Directory caches are the normal Tor server nodes which download the Tor node directory and serve the mirror of directory authority to provide Tor server information for other Tor server and client nodes. Commonly most of Tor client and server nodes download Tor server node list and information from directory caches instead of directory authorities. It is very easy for most Tor server nodes to be directory caches, so directory caches decrease the download burden greatly for directory authorities.

In general, there are 4 types of nodes on Tor anonymous network, directory authority, directory cache, server node, and client node.

Tor network principle

After we have the overall knowledge for Tor network circuit and Tor network node classification. We will discuss deeper principle of Tor network. Here let's take the http website browse example again.

When a client Tor node, installed Tor client application, wants to visit a website anonymously using Tor network, the client Tor node first needs to set up a Tor circuit with 3 mediate Tor server nodes. After this circuit is setup, the client node can visit website through

this circuit. In other words, the client node can transmit one or more http, or any other TCP data stream through the circuit. Before the client node can set up one valid Tor circuit, this client node must get at least 1/4 running Tor server nodes' information and get a live consensus status file from one of two kinds of directory servers, directory authority or directory cache. Here consensus status file includes the list of almost all valid Tor server nodes and their basic information. Normally client nodes download fresh consensus status file from directory caches if they know any directory cache. If it is the first time for client nodes to download the consensus status file or if the client nodes do not know any directory cache, the client nodes will download consensus status file from directory authority directly.

How does consensus status file form? To know this, we need to talk about the descriptor of Tor server nodes. Every Tor server node uploads its descriptor to directory authorities. The descriptor includes detailed information about this Tor server node such as Nick name, IP address, publish time, and so on. When one or more of the following 3 situations appear, Tor server nodes must upload their descriptors to directory authorities that they know.

1. There is 18-hours interval since last generated descriptor.

2. Except of changes of bandwidth and uptime, there are other fields changed.

3. There are more than +/- 50% change for bandwidth since last generated descriptor and at least 20 minutes interval since last generated descriptor.

4. The uptime of the server node is reset by restarting.

After receiving server nodes' descriptors, about every 15 minutes the directory authorities will vote a consensus status with the agreement of every directory authority. This consensus status includes the list and key information of all server nodes on Tor network.

Every consensus status document has three times that are valid-after time, fresh-until time, and valid-until time in early to late order. When there is new consensus status appearing, all the nodes including directory caches, server nodes, and client nodes, need to download it.

Directory caches download a new consensus status under any of following conditions.

1. There is no consensus status document for this directory cache.

2. The consensus status document is out of valid-until time.

Except of above 2 conditions, the directory caches download new consensus status file at a random time to avoid the download congestion just after the new consensus status file comes out.

We mentioned that one of 2 conditions that lead to setting up a circuit by a client node is getting at least 1/4 Tor server nodes' information. This means getting 1/4 descriptors of all running Tor server nodes. Every Tor node not only has a live consensus status file but also has a file including all server nodes' descriptors it knows. These descriptors are downloaded from directory authorities or directory caches. All nodes compare their owned descriptors to the ones in the latest consensus status. If there are at least 16 new descriptors and at least 10 minutes since last download, the nodes will request a new download for descriptors. Normally, client and server nodes download new descriptors from directory caches. Directory caches download new descriptors from directory authorities. And directory authorities download new descriptors among each other.

After qualifying the conditions for setting up a Tor circuit, how do client nodes select the correct Tor server nodes? There are the following restrictions for the selection of server nodes:

" - Clients SHOULD NOT use non-'Valid' or non-'Running' routers unless

requested to do so.

- Clients SHOULD NOT use non-'Fast' routers for any purpose other than very-low-bandwidth circuits (such as introduction circuits).
- Clients SHOULD NOT use non-'Stable' routers for circuits that are Likely to need to be open for a very long time (such as those used for IRC or SSH connections).
- Clients SHOULD NOT choose non-'Guard' nodes when picking entry guard nodes.
- Clients SHOULD NOT download directory information from non-'V2Dir' caches." [8]

The explanation about different node flags is as follows:

'"Authority" if the router is a directory authority.

"BadExit" if the router is believed to be useless as an exit node (because its ISP censors it, because it is behind a restrictive proxy, or for some similar reason).

"BadDirectory" if the router is believed to be useless as a directory cache (because its directory port isn't working, its bandwidth is always throttled, or for some similar reason).

"Exit" if the router is more useful for building general-purpose exit circuits than for relay circuits. The path building algorithm uses this flag; see path-spec.txt.

"Fast" if the router is suitable for high-bandwidth circuits.

"Guard" if the router is suitable for use as an entry guard.

"HSDir" if the router is considered a v2 hidden service directory.

---

[8] Tor directory protocol, version 3, https://www.torproject.org/svn/trunk/doc/spec/dir-spec.txt

"Named" if the router's identity-nickname mapping is canonical, and this authority binds names.

"Stable" if the router is suitable for long-lived circuits.

"Running" if the router is currently usable.

"Valid" if the router has been 'validated'.

"V2Dir" if the router implements the v2 directory protocol.

"V3Dir" if the router implements this protocol.'[9]

Tor software introduction

Tor software can run on multiple platforms such as Windows, Mac, Linux, and Unix. Commonly Tor is used with Provoxy, another proxy program that can avoid address disclosure when some applications on Tor users' computers go to DNS server on the Internet to resolve the hostname. This IP address resolution reveals the Tor user's IP address and its destination IP address. So it will disclose the address and location information to sniffers. Privoxy is a proxy server to solve this problem. It has the ability to keep the privacy.

After installation of Tor, we need to configure the Internet browser and other applications that use Tor network such as instant messaging application. The main setup work is that we configure applications to use Tor port to enter Tor network.

We can install a bundle installation including Tor, Provoxy, and other useful applications. Vadalia is the installation bundle of Tor. It is a cross-platform GUI for Tor control. Like the name of Vadalia, one kind of onion, it supports Tor, second-generation

---

[9] Tor directory protocol, version 3, https://www.torproject.org/svn/trunk/doc/spec/dir-spec.txt

onion router. With Vidalia Tor users can easily start, stop, and see all status information about Tor use.

We can go to Tor project website to download installation programs.

http://www.torproject.org/download.html.en#Warning

Platform	Download Stable
Windows Browser Bundle (Contains Tor, Torbutton, Polipo, and Firefox)	
Windows (Contains only Tor) *Vista, XP, 2000, 2003 Server, Millenium, 98SE*	0.1.2.19 (sig)
Mac (Contains Tor, Torbutton, and Privoxy) *Universal Binary (OSX 10.4 & 10.5)*	0.1.2.19 (sig)
Mac (Contains Tor, Torbutton, and Privoxy) *PowerPC Only (OSX 10.3)*	0.1.2.19 (sig)
Linux/Unix packages (Contains only Tor) *Redhat/CentOS, Fedora, Debian, Ubuntu, SUSE*	Linux/Unix download
Source tarballs `./configure && make && src/or/tor`	0.1.2.19 (sig)

Figure 6 Tor installation download [10]

After download, you can install Tor bundle easily.

[10] http://www.torproject.org/download.html.en

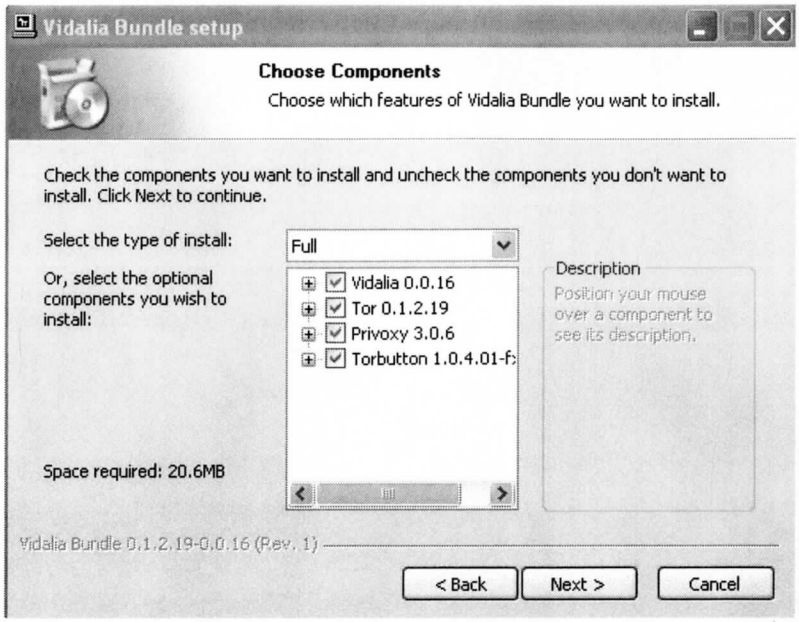


Figure 7 Tor bundle installation step 1 [11]

From above installation figure, we can see

After installation, we need to press Torbutton at the right-bottom corner of Firefox to trigger Tor network.

[11] Captured from Vadalia software

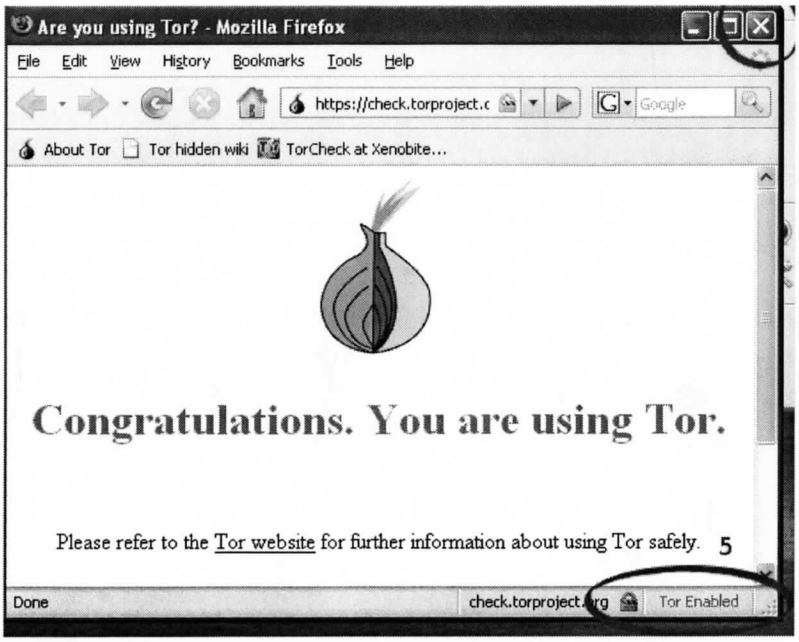


Figure 8 Torbutton on Firefox browser [12]

The Torbutton appears green "Tor Enabled" to show Tor is working. Pressing it again will switch Tor to stop running.

We can use Vadalia to see the Tor network status and configure Tor conveniently.

---

[12] http://www.torproject.org/torbrowser/index.html.en

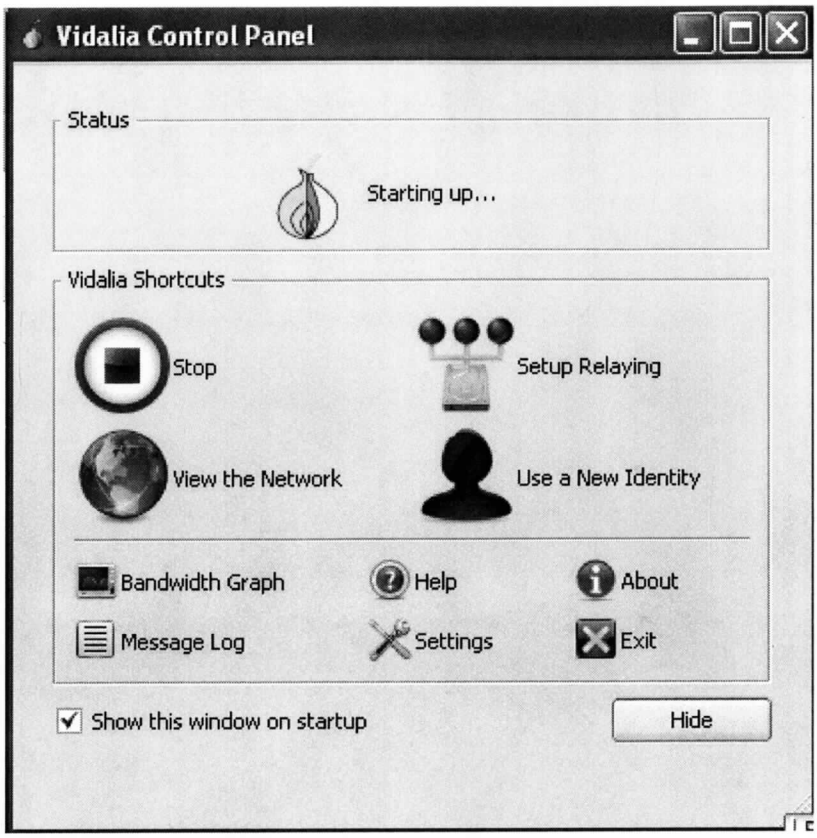


Figure 9 Vadalia Control Center [13]

From Vadalia control center, we can see Tor user's bandwidth graph, real-time Tor network map, Tor user's message log, and Tor software running status. We also can enter Tor setup panel with clicking "settings" button on Vadalia control center.

The followings are the figures of Vadalia monitor function for Tor network.

[13] Captured from Vadalia software

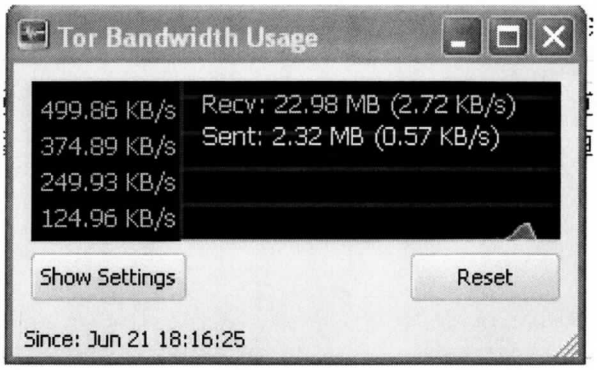


Figure 10 Tor user's Bandwidth Graph [14]

Above graph shows real-time Tor user's input and output data bandwidth information.

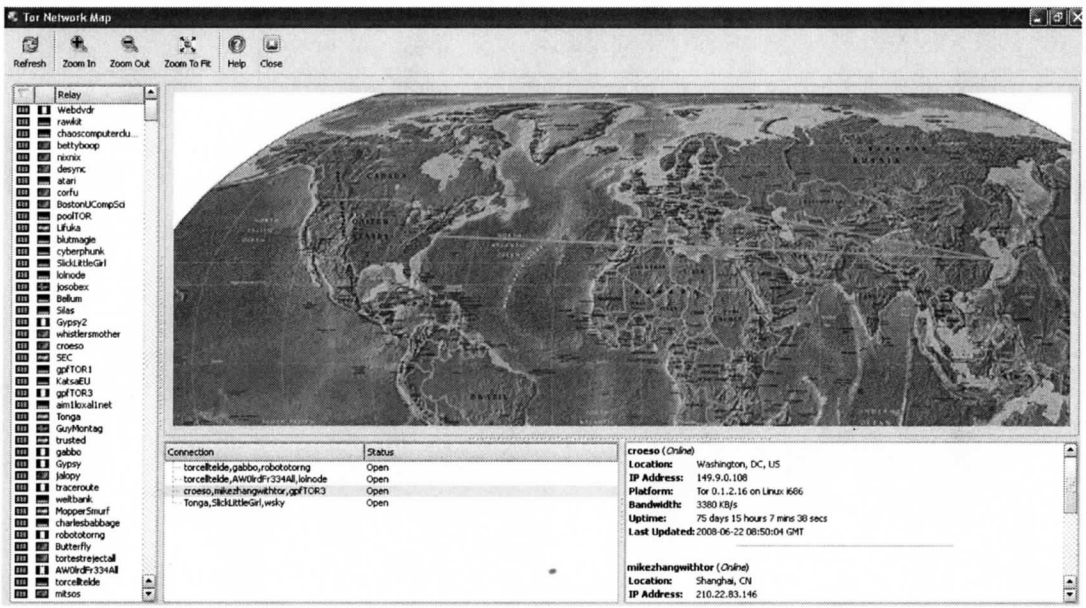


Figure 11 Real-time Tor network map [15]

Above real-time Tor network map shows the geography world map with red points for Tor users' distribution, and green lines for this Tor user's circuit. Under the map, there are all

---

[14] Captured from Vadalia software

[15] Captured from Vadalia software

circuits that this Tor user is setting up. The detail for every circuit is showed at the right-bottom corner. At the left side of this figure is the list of server nodes that this Tor user has.

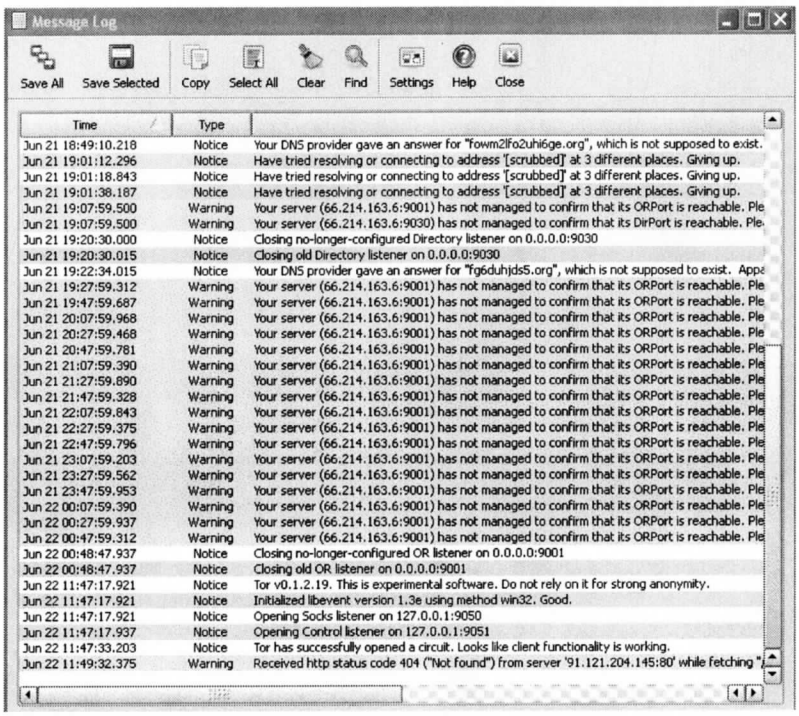


Figure 12 Tor message log [16]

Above is the log center for requested messages.

Figure 13 Tor server nodes information [17]

As showed at left side, there are bars and country flags in front of server nodes' names. Bar means the speed of this Tor server node. More bars, faster speed. The country flags show the geography location of server nodes.

[16] Captured from Vadalia software

[17] Captured from Vadalia software

Figure 14 Tor network check [18]

When we run Tor, we need to confirm that we actually go surfing through Tor anonymous network. We can go to some websites that check Tor running status. For example go to https://check.torproject.org/ to check it. We will get above screen if we use Tor successfully.

Tor network performance test and analysis

Through analysis of Tor network principle and structure, I set up an experiment Tor network system to test the performance of Tor.

---

[18] http://www.torproject.org/torbrowser/index.html.en

I downloaded a data file six hundred times from one website both with Tor network and without Tor network. The following figure is the result of download.

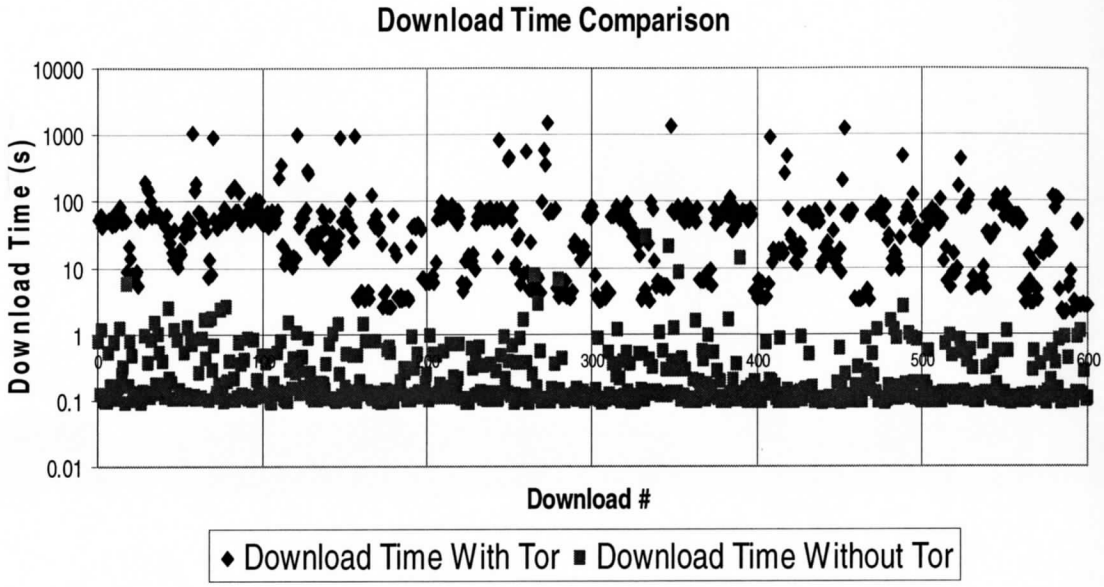


Figure 15 Performance test of existing Tor network

The horizontal ordinate is the download time. The ordinate is the download number. In the figure, the pink points mean each download time without Tor. The blue points are the download times using Tor network. We can see that the time distribution without Tor network is very lower and more even than that of time with Tor network. With our preliminary research for Tor software code, we found that it is not very reasonable for Tor software to select the server nodes to set up the Tor circuit. Some of selected server nodes have low bandwidth and are not stable enough. It impacts the speed and stability of Tor network. These deficiencies affect the wide use of Tor network.

With above description, we had the comprehensive knowledge about Tor. We can get to know the principle of Tor anonymous network and detailed information about Tor running.

Also we know some deficiencies of Tor network. These are the basis for visualization of Tor network. Next we will discuss visualization of Tor network.

Visualization for Tor network

Tor network visualization overview

The principle and structure of Tor network is a little complicated for many Internet users. It is an obstacle for wide use of Tor network. In order to let more Internet users get to know the principle and knowledge about Tor anonymous communication network, in my project I designed a visualization system for Tor anonymous network. This system includes a background database of all known Tor server nodes and a website for showing the information of that background database and retrieving the information about some Tor server nodes.

I used MySQL to set up the background database. MySQL is a robust and fast relational database management system (RDBMS). It is an excellent open source RDBMS and permits multiple users and multiple threads. I used MySQL to set up this database successfully and efficiently.

I used HTML, CSS, and PHP languages to set up the information website. PHP is a free and powerful server-side scripting language. It can be embedded inside HTML code conveniently. It is one wide-used server-side scripting language.

Through our database and website, we can get better visualization for Tor network.

I used the Fedora Linux operating system to host the database and the website. Fedora is an excellent and stable Linux operation system. It is an open source operating system that is very suited for a test environment.

For the website publish, I selected the Apache server as the web server application. Apache server is simple but very powerful.

With these four great applications, which are also called LAMP combo, we can set up Tor visualization system easily.

Tor network visualization database

First let's take a look at the background database for Tor server nodes.

This database has three tables whose names are Tor_nodes, Tor_authorities, and Tor_script_statues.

```
mysql> show tables;
+----------------------+
| Tables_in_torsearch |
+----------------------+
| tor_authorities      |
| tor_nodes            |
| tor_script_status    |
+----------------------+
```

Figure 16 Database table

Table Tor_nodes is the table for all Tor server nodes. Table Tor_authorities is for the Tor directory authorities. Table Tor_script_status is for the running status of update script of database.

Tor_nodes table has 35 fields that are Keep, AuthorityID, IdentityHash, ID, LineNumber, NickName, DescriptorHash, PublicationDate, PublicationTime, IP, ORPort, DIRPort, FQDN, NodeVersion, ORPort_active, DIRPort_active, CountryCode, WhoIsIP, IsAuthority, IsExit, IsFast, IsGuard, IsNamed, IsStable, IsRunning, IsValid, IsV2Dir, IsBadExit, IsBadDirectory, IP1, IP2, IP3, IP4, HasWebservice, IsFakeFQDN. Among them, ID is the primary key for the table.

Field	Type	Null	Key	Default	Extra
Keep	varchar(10)	YES		NULL	
AuthorityID	varchar(30)	YES		NULL	
IdentityHash	varchar(550)	YES		NULL	
ID	int(11)	NO	PRI	NULL	auto_increment
LineNumber	int(11)	YES		NULL	
NickName	varchar(150)	YES		NULL	
DescriptorHash	varchar(550)	YES		NULL	
PublicationDate	date	YES		NULL	
PublicationTime	time	YES		NULL	
IP	varchar(50)	YES		NULL	
ORPort	int(11)	YES		NULL	
DIRPort	int(11)	YES		NULL	
FQDN	varchar(50)	YES		NULL	
NodeVersion	varchar(50)	YES		NULL	
ORPort_active	varchar(50)	YES		NULL	
DIRPort_active	varchar(50)	YES		NULL	
CountryCode	varchar(10)	YES		NULL	
WhoIsIP	varchar(50)	YES		NULL	
IsAuthority	int(11)	YES		NULL	
IsExit	int(11)	YES		NULL	
IsFast	int(11)	YES		NULL	
IsGuard	int(11)	YES		NULL	
IsNamed	int(11)	YES		NULL	
IsStable	int(11)	YES		NULL	
IsRunning	int(11)	YES		NULL	
IsValid	int(11)	YES		NULL	
IsV2Dir	int(11)	YES		NULL	
IsBadExit	int(11)	YES		NULL	
IsBadDirectory	int(11)	YES		NULL	
IP1	varchar(10)	YES		NULL	
IP2	varchar(10)	YES		NULL	
IP3	varchar(10)	YES		NULL	

Figure 17 Tor_nodes table structure

Tor_authorities table has 19 fields that are Changed, ID, FileName, FileSize, FileTime, FileContent, NickName, StatusVersion, DirSource, Fingerprint, Contact, Published,

DirOptions, ClientVersions, ServerVersions, RouterCount, and DirSignature. ID is the primary key.

Field	Type	Null	Key	Default	Extra
Changed	varchar(10)	YES		NULL	
ID	int(11)	NO	PRI	NULL	auto_increment
FileName	varchar(100)	YES		NULL	
FileSize	varchar(50)	YES		NULL	
FileTime	varchar(100)	YES		NULL	
FileContent	longtext	YES		NULL	
NickName	varchar(100)	YES		NULL	
StatusVersion	varchar(50)	YES		NULL	
DirSource	varchar(50)	YES		NULL	
Fingerprint	varchar(250)	YES		NULL	
Contact	varchar(50)	YES		NULL	
Published	datetime	YES		NULL	
DirOptions	varchar(50)	YES		NULL	
ClientVersions	varchar(150)	YES		NULL	
ServerVersions	varchar(150)	YES		NULL	
RouterCount	int(11)	YES		NULL	
DirSignature	varchar(950)	YES		NULL	
Scanning	int(11)	YES		NULL	
DirSigningKey	varchar(950)	YES		NULL	

Figure 18 Tor_authorites table structure

Tor_script_status table has four fields that are IsRunning, Object, LastRun, and Message.

Field	Type	Null	Key	Default	Extra
IsRunning	varchar(10)	YES		NULL	
Object	varchar(30)	YES		NULL	
LastRun	time	YES		NULL	
Message	varchar(50)	YES		NULL	

Figure 19 Tor_script_status

The database is updated using a PHP script. I used crontab in Linux to execute this PHP script every one hour. I used "crontab - e" command to enter the edit mode to create and edit our automatically executive script. I wrote the following code inside the crontab file:

0 * * * * php /var/www/html/torhourly.php

This means that the script of torhourly.php can execute automatically every hour. The first "0" in the code means each 0 minute. The second to the fifth place are all "*" marks that mean any hour, any day, any month, and any day of a week. So the whole code is to execute automatically "torhourly.php" script under /var/www/html/ directory.

I used some of ideas and codes from https://torstat.xenobite.eu as reference for our script codes.

Next let's take a look at the detail of torhourly.php script.

First, the script includes some other important PHP scripts that are for time, connecting database, script status, checking country code, and so on.

```
1 <?php
2
3     include('f_MicroTimeFloat.inc.php');
4     include('f_ConnectDB.inc.php');
5     include('f_ScriptStatus.inc.php');
6     include('f_CheckCC.inc.php');
7     include('f_CheckFQDN.inc.php');
```

Figure 20 PHP script part 1[19]

After including all other PHP scripts, we recall a user's time function, MicroTimeFloat(), to return the accurate time. We connect to database with right user and password. We update the script status table to show the beginning of this PHP script.

[19] Reference from https://torstat.xenobite.eu

```
9     $ScriptObject = "Torhously";
10
11    //
      ----------------------------------------
      ---------------------
12
13    $MicroTimeStart = MicroTimeFloat();
14
15    ConnectDB();
16
17    ScriptStatus($ScriptObject,1,"...");
```

Figure 21 PHP script part 2 [20]

When we run Tor software as client or server mode, the consensus status files are stored locally. The information of these consensus status files can be imported into our database with PHP script. Before importing these consensus status files, we need to index consensus status files.

```
24    // index cached status dir
25    $DirName = "/Tor/cached-status";
26    $DirIdx = opendir($DirName);
27    $i = 0;
28    while($FileName = readdir($DirIdx)) {
29  if($FileName != "." && $FileName != "..") {
30      $FileList[$i] = $FileName.",".filesize($DirName."/".$FileName).",".filemtime(
      $DirName."/".$FileName);
31      $i++;
32  }
33    }
```

Figure 22 PHP script part 3 [21]

Directory of "/Tor/cached-status" is the place for storing consensus files. We open this directory and read each file, then put file name, file size, and file time together into an array variable "$FileList". Every element of $FileList is composed of file name, file size, and file modified time linked by comma.

---

[20] Reference from https://torstat.xenobite.eu

[21] Reference from https://torstat.xenobite.eu

```
36      // clear changed flag
37      @mysql_query("UPDATE tor_authorities SET Changed=0");
38      // compare times
39      reset($FileList);
40      while(list($Key,$Value) = each($FileList)) {
41        list($FileName,$FileSize,$FileTime) = explode(",",$Value);
42    $Row_tor_authorities = @mysql_fetch_array(@mysql_query("SELECT ID,FileName,FileTime
  FROM tor_authorities WHERE FileName='".$FileName."' ORDER BY FileTime DESC"));
43    if(@mysql_affected_rows() != 0) {
44        // update authority
45        if($FileTime != $Row_tor_authorities['FileTime']) {
46      @mysql_query("UPDATE tor_authorities SET FileName='".$FileName."',FileSize='".
  $FileSize."',FileTime='".$FileTime."',Changed='1',FileContent='".file_get_contents(
  $DirName."/".$FileName)."' WHERE FileName='".$FileName."'");
47        }
48    } else {
49        // new authority
50        @mysql_query("INSERT tor_authorities SET FileName='".$FileName."',FileSize='".
  $FileSize."',FileTime='".$FileTime."',Changed='1',FileContent='".file_get_contents(
  $DirName."/".$FileName)."'");
51    }
52      }
53      // update nescessary?
54      $Result_tor_authorities = @mysql_query("SELECT * FROM tor_authorities WHERE
  Changed='1'");
55      if(@mysql_affected_rows() == 0) {
56    ScriptStatus($ScriptObject,"(no change)");
57    @mysql_close();
58    exit();
59      }
```

Figure 23 PHP script part 4 [22]

Every consensus status file is from a directory authority. So we can regard each consensus status file as one row of directory authority in the Tor_authorities table. There is a field named "changed" for showing whether there is any change for directory authorities after every update of the database. When value of "changed" field is 0, it means that there is no any change for that directory authority. If its value is 1, it means there is any change for that directory authority. First we need to reset "changed" field to 0.

We first judge if there is that authority in the table. If there is no that authority/consensus status, we create new row for this new authority/consensus status in Tor_authorities table and write 1 for "changed" field. If there is that authority in table, we

[22] Reference from https://torstat.xenobite.eu

judge if the old value of "file time" field in the table is the same as the file time of the local consensus status file. If they are different, we update this authority in the table and set 1 for "changed" field.

After judgment, we check whether there is any authority whose "changed" field is 1. If there is no any 1 for "changed" field, the update script will stop execute and close and we will not update database.

When execute this part of script, we update the "FileName", "FileSize", "FileTime", and "FileContent" fields of authorities. Among them, we will write all content of local consensus status into "FileContent" field.

```
61      ### process files ###
62
63      // parser
64      while($Row_tor_authorities = @mysql_fetch_array($Result_tor_authorities)) {
// archive directory
65    $File = "/Tor/archive/".date("Y-m-d-H-i-s",$Row_tor_authorities['FileTime'])."_"
.$Row_tor_authorities['FileName']."_".$Row_tor_authorities['NickName'].".asc";
66
67    $Handle = fopen($File,"w");
68    fwrite($Handle,$Row_tor_authorities['FileContent']);
69    fclose($Handle);
70
71    @mysql_query("UPDATE tor_authorities SET Scanning='1' WHERE ID='".
$Row_tor_authorities['ID']."'");
72    @mysql_query("UPDATE tor_nodes SET Keep='0' WHERE AuthorityID='".
$Row_tor_authorities['ID']."'");
73    $FileContentArray = explode("\n",$Row_tor_authorities['FileContent']);
74    $FileContentLines = count($FileContentArray);
```

Figure 24 PHP script part 5 [23]

After confirming that there is any authority including some new information, we begin to process local consensus status file.

We use a "while" loop to process every authority's consensus status file. First we archive the consensus status files to directory of "/Tor/archive/". After that we begin to

---

[23] Reference from https://torstat.xenobite.eu

process every consensus status file one by one with the beginning "while" loop. We mark 1 for "Scanning" filed of every processing authority. We mark 0 for "keep" field of every node that means before processing we will not keep any server nodes in Tor_nodes table. After processing the consensus status file, we will change "keep" field to 1 for those nodes that need to keep. We will discuss this point later. At last, we divide all lines of consensus status file and calculate how many lines it has.

Here we need to take a look at the structure of consensus file.

```
network-status-version 2
dir-source tor.dizum.com 194.109.206.212 80
fingerprint 7EA6EAD6FD83083C538F44038BBFA077587DD755
contact 1024R/8D56913D Alex de Joode <adejoode@sabotage.org>
published 2008-04-20 07:29:48
dir-options
dir-signing-key
-----BEGIN RSA PUBLIC KEY-----
MIGJAoGBAL7QJ6cmXhMlexV97ehnV5hn5ePOeo0sbDYXhlfw52CheEycouqSD9Y/
3qEo0Rm7XTEol0dRW34ca1LMIXGM4B4whXxBKCRRYe1RY6nF70zb2EUuaHWEWc+f
c6JWYUWZSPpW1uyjyLPUI/ikyyH7zmtR4MfhSeNdt2zSakojYNaPAgMBAAE=
-----END RSA PUBLIC KEY-----
r Unnamed ACBBvR4J0XWDAGhx2WOjiLmW1d0 ftOpFw3CGwQXrkcWyQorr6oLDHs
  2008-04-19 15:32:19 91.64.41.141 9001 9030
s Exit V2Dir Valid
opt v Tor 0.1.2.18
r torAdm84srv001 ACHpOeZYOuoyfLa5KmO65QDgN4M 7VklacI8Yga6glPr0zX8kQIzdo4
  2008-04-20 04:58:23 194.102.180.118 9001 0
s Exit Fast Running Stable Valid
opt v Tor 0.2.0.23-rc (r14173)
r Unnamed AHKt6dPpMD1D/MFTmNLpY6ni0yM RQ3sIUkj4t0+2yGMAsMUGhhHVvY
  2008-04-20 07:05:06 140.116.104.15 443 9030
s Exit Fast Running V2Dir Valid
opt v Tor 0.1.2.19
r bigbear3001 AMVDhgqAERZG1MYjq9NwUtmknas yFEeCfzTGyHH1yd7us6TdHDukEM
  2008-04-20 06:00:27 88.198.72.170 9001 9030
s Exit Fast Guard Running Stable V2Dir Valid
opt v Tor 0.1.2.18
r attilator2 ANeT1iLJoSM20J2bAlog11T3VDE qhVcn5tsvzc985B21YyCvJgjwEA
  2008-04-20 00:52:11 75.91.80.162 9001 9030
s V2Dir Valid
opt v Tor 0.1.2.19
```

```
r Beaver /96bKo4soysolMgKn5Hex2nyFSY BuCDxpJB83Ew2d2whhBz8E+m9tg
  2008-04-19 14:47:40 80.223.84.180 443 0
s Fast Running Valid
opt v Tor 0.2.1.0-alpha-dev (r14297)
directory-signature dizum
-----BEGIN SIGNATURE-----
iMbE5C1MsSgItn+86ND02EFzvXHZEOpHGXx1BX9xSThFTzb4ADXeh7fDofK535a7
mNwAri/e19lp380Uj5TzBQEb40mPwmuGRgc6M+CzAJASCK52crBKE88Dg5O1JW/2
3mtNWxJmn3SsvSXI4ZiRCv26cEoeKuhsUIDyudH6h58=
-----END SIGNATURE-----
```

Figure 25 The structure of consensus status file

Consensus status file include 3 parts that are the beginning authority's information, the list of server nodes, and the end authority's information. The beginning authority information includes the basis information of this authority who signs this consensus status file. The

second part is the basis information of all server nodes included in this consensus status file. Every node's information has 3 lines. The first line begins with "r" and has the basic information about this node. The second line begins with "s" and shows all flags that this node has. The third line begins with "opt" and shows the Tor software version. The end part of the consensus status file is the directory authority's name and signature.

```
$i = 0;
while($i < $FileContentLines) {
    $Line = $FileContentArray[$i];
$aaa = strpos($Line," ");
if ($aaa === false) {
$Keyword = substr($Line, 0, strlen($Line));
}
else {
  $Keyword = substr($Line,0,$aaa);
}

    switch($Keyword) {
  case "network-status-version":
      $StatusVersion = trim(strstr($Line," "));
      break;
  case "dir-source":
      $DirSource = trim(strstr($Line," "));
      break;
  case "fingerprint":
      $Fingerprint = trim(strstr($Line," "));
      break;
  case "contact":
      $Contact = trim(strstr($Line," "));
      break;
  case "published":
      $Published = trim(strstr($Line," "));
      break;
  case "dir-options":
      $DirOptions = trim(strstr($Line," "));
      break;
  case "client-versions":
      $ClientVersions = trim(strstr($Line," "));
      break;
  case "server-versions":
      $ServerVersions = trim(strstr($Line," "));
      break;
  case "dir-signing-key":
      $DirSigningKey = $FileContentArray[$i + 2]."\n".$FileContentArray[$i + 3]."\n".
$FileContentArray[$i + 4];
      $i = $i + 5;
      break;
  case "directory-signature":
      $AuthNickName = trim(strstr($Line," "));
      $DirSignature = $FileContentArray[$i + 2]."\n".$FileContentArray[$i + 3]."\n".
$FileContentArray[$i + 4];
      $i = $i + 5;
      break;
```

Figure 26 PHP script part 6 [24]

[24] Reference from https://torstat.xenobite.eu

We read the first word of every line of consensus status file and assign the first word to the variable $Keyword. The above codes get the information for the authority signing this consensus status and assign the information to suitable variables such as $AuthNickName, $Contact, $Published, and so on. Here we use a "switch" structure to judge and assign keyword to suitable variables.

```
121     case "r":
122         // first line
123         $LineNumber = $i;
124         $ValuesArray = explode(" ",$Line);
125         $NickName = $ValuesArray[1];
126         $IdentityHash = $ValuesArray[2];
127         $DescriptorHash = $ValuesArray[3];
128         $PublicationDate = $ValuesArray[4];
129         $PublicationTime = $ValuesArray[5];
130         $IP = $ValuesArray[6];
131         $ORPort = $ValuesArray[7];
132         $DIRPort = $ValuesArray[8];
133
134         unset($ValuesArray);
135         $i++;
```

Figure 27 PHP script part 7 [25]

After read the authority's information, the script begins to process the nodes' information. With "case 'r'" code as the sign for first line of node' information, the script splits all information of node to separate element of an array variable by "explode" function. Then assign the elements of this array to suitable variables. At the end part of above code, we use "$i++" to step to the next line of consensus status file.

[25] Reference from https://torstat.xenobite.eu

```
136         // router flags
137         // second line
138         // predefine zero
139         $FlagsArray = array("Authority","Exit","Guard","Named","Stable","Running",
    "Valid","V2Dir","Fast","BadExit","BadDirectory");
140         while(list($Key,$Flag) = each($FlagsArray)) {
141       ${"Is".$Flag} = 0;
142         }
143         reset($FlagsArray);
144
145         // scan for flags
146         $Line = $FileContentArray[$i];
147         $ValuesArray = explode(" ",$Line);
148         while(list($Key,$Value) = each($ValuesArray)) {
149       switch($Value) {
150           case "Fast":
151         $IsFast = 1;
152         break;
153           case "Exit":
154         $IsExit = 1;
155         break;
156           case "Authority":
157         $IsAuthority = 1;
158         break;
159           case "Guard":
160         $IsGuard = 1;
161         break;
162           case "Named":
163         $IsNamed = 1;
164         break;
165           case "Stable":
166         $IsStable = 1;
167         break;
168           case "Running":
169         $IsRunning = 1;
170         break;
171           case "Valid":
172         $IsValid = 1;
173         break;
174           case "V2Dir":
175         $IsV2Dir = 1;
176         break;
177           case "BadExit":
178         $IsBadExit = 1;
179           case "BadDirectory":
180         $IsBadDirectory = 1;
181       }
182         }
183         $i++;
```

Figure 28 PHP script part 8 [26]

[26] Reference from https://torstat.xenobite.eu

After processing the first line of node's information, the script begins to read the flag information using the combination of "while" and "switch" functions. After above codes, we get and assign node's flag information to suitable variables.

```
185         // options
186         // third line
187         $Line = $FileContentArray[$i];
188         $ValuesArray = explode(" ",$Line);
189         next($ValuesArray);
190         next($ValuesArray);
191         unset($NodeVersion);
192         while(list($Key,$Value) = each($ValuesArray)) {
193       $NodeVersion = $NodeVersion." ".$Value;
194         }
195         $NodeVersion = trim($NodeVersion);
```

Figure 29 PHP script part 9 [27]

Above codes are to process the third line for node's information that is the version of Tor software.

[27] Reference from https://torstat.xenobite.eu

```
204         // check if router already in DB
205         $Result_tor_nodes = @mysql_query("SELECT ID FROM tor_nodes WHERE
    IdentityHash='".$IdentityHash."' AND AuthorityID='".$Row_tor_authorities['ID']."'"
    );
206         if(@mysql_num_rows($Result_tor_nodes) != 0) {
207       // UPDATE ROUTER
208       unset($FlagsSQLQuery);
209       while(list($Key,$Flag) = each($FlagsArray)) {
210           $FlagsSQLQuery .= "Is".$Flag."='".${"Is".$Flag}."',";
211       }
212       $SQL = ("UPDATE tor_nodes SET LineNumber='".$LineNumber."',
213           NickName='".$NickName."',
214           DescriptorHash='".$DescriptorHash."',
215           PublicationDate='".$PublicationDate."',
216           PublicationTime='".$PublicationTime."',
217           IP='".$IP."',
218           IP1='".$IP1."',
219           IP2='".$IP2."',
220           IP3='".$IP3."',
221           IP4='".$IP4."',
222           ORPort='".$ORPort."',
223           DIRPort='".$DIRPort."',
224           FQDN='".$FQDN."',
225           NodeVersion='".$NodeVersion."',
226           IsAuthority='".$IsAuthority."',
227           IsExit='".$IsExit."',
228           IsFast='".$IsFast."',
229           IsGuard='".$IsGuard."',
230           IsNamed='".$IsNamed."',
231           IsStable='".$IsStable."',
232           IsRunning='".$IsRunning."',
233           IsValid='".$IsValid."',
234           IsV2Dir='".$IsV2Dir."',
235           IsBadExit='".$IsBadExit."',
236           IsBadDirectory='".$IsBadDirectory."',
237           ORPort_active='9',
238           DIRPort_active='9',
239           Keep='1' WHERE IdentityHash='".$IdentityHash."' AND AuthorityID='".
    $Row_tor_authorities['ID']."'"
240       );
241       @mysql_query($SQL);
242         } else {
243       // INSERT ROUTER
```

Figure 30 PHP script part 10 [28]

After the script finish reading information about node, it will judge whether this node is in the database. If the node is in the database, the script will update the fields in the Tor_nodes table for this node. If the node is a new one that is not in the database, the script will insert it to Tor_nodes table. In above codes, we change the value of "keep" field of updated or new-inserted authorities to 1.

[28] Reference from https://torstat.xenobite.eu

```
253         $CountryCode = CheckCC($IP);
254         // $CountryCode = "-";
255
256         unset($FlagsSQLQuery);
257         while(list($Key,$Flag) = each($FlagsArray)) {
258             $FlagsSQLQuery .= "Is".$Flag."='".${"Is".$Flag}."',";
259         }
260         $SQL = ("INSERT tor_nodes SET LineNumber='".$LineNumber."',
261             AuthorityID='".$Row_tor_authorities['ID']."',
262             NickName='".$NickName."',
263             IdentityHash='".$IdentityHash."',
264             DescriptorHash='".$DescriptorHash."',
265             PublicationDate='".$PublicationDate."',
266             PublicationTime='".$PublicationTime."',
267             IP='".$IP."',
268             IP1='".$IP1."',
269             IP2='".$IP2."',
270             IP3='".$IP3."',
271             IP4='".$IP4."',
272             ORPort='".$ORPort."',
273             DIRPort='".$DIRPort."',
274             FQDN='".$FQDN."',
275             CountryCode= '".$CountryCode."',
276             NodeVersion='".$NodeVersion."',
277             IsAuthority='".$IsAuthority."',
278             IsExit='".$IsExit."',
279             IsFast='".$IsFast."',
280             IsGuard='".$IsGuard."',
281             IsNamed='".$IsNamed."',
282             IsStable='".$IsStable."',
283             IsRunning='".$IsRunning."',
284             IsValid='".$IsValid."',
285             IsV2Dir='".$IsV2Dir."',
286             IsBadExit='".$IsBadExit."',
287             IsBadDirectory='".$IsBadDirectory."',
288             ORPort_active='9',
289             DIRPort_active='9',
290             Keep='1'"
291         );
292         @mysql_query($SQL);
```

Figure 31 PHP script part 11 [29]

Above codes are to insert a new node.

[29] Reference from https://torstat.xenobite.eu

```
302   // delete old routers
303   @mysql_query("DELETE FROM tor_nodes WHERE AuthorityID='".$Row_tor_authorities
    ['ID']."' AND Keep='0'");
304
305   // count routers
306   @mysql_query("SELECT ID FROM tor_nodes WHERE AuthorityID='".
    $Row_tor_authorities['ID']."'");
307   $RouterCount = @mysql_affected_rows();
308   $Sum = $Sum + $RouterCount;
309
310   // update authority information
311   $SQL = ("UPDATE tor_authorities SET StatusVersion='".$StatusVersion."',
312       DirSource='".$DirSource."',
313       Fingerprint='".$Fingerprint."',
314       Contact='".$Contact."',
315       Published='".$Published."',
316       DirOptions='".$DirOptions."',
317       ClientVersions='".$ClientVersions."',
318       ServerVersions='".$ServerVersions."',
319       NickName='".$AuthNickName."',
320       RouterCount='".$RouterCount."',
321       DirSignature='".$DirSignature."',
322       Scanning='0',
323       DirSigningKey='".$DirSigningKey."' WHERE ID='".$Row_tor_authorities['ID'
    ]."'
324   ");
325   @mysql_query($SQL);
326     }
327
328     #### END ###
329     // stop timer
330     $MicroTimeStop = MicroTimeFloat();
331     $MicroTimeDiff = round($MicroTimeStop - $MicroTimeStart,0);
332     ScriptStatus($ScriptObject,0,"Last: ".date("H:i:s",$MicroTimeDiff)."h
    (Processed: ".$Sum." Nodes)");
333     echo "update successfully!";
334     @mysql_close();
335     exit();
336 ?>
```

Figure 32 PHP script part 12 [30]

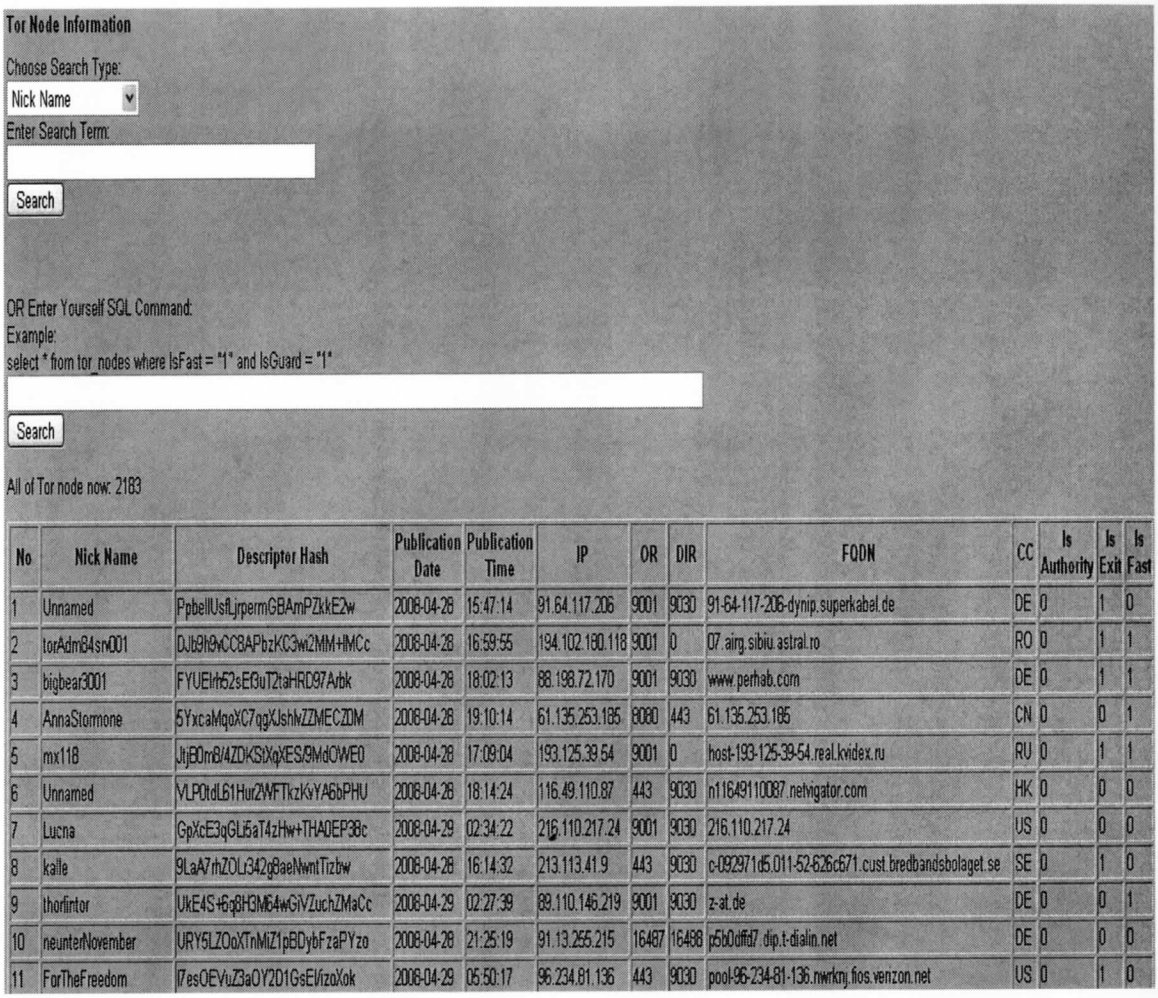
After reading all consensus status files under /Tor/cached-status directory, we update all nodes in the Tor_nodes table. We will delete those server nodes with no changes in the Tor_nodes table. Then calculate the amount of all nodes in Tor_nodes table and update the new information of authorities in Tor_authorities table. At the end of the script, we stop the script and exit. All update work to database is finished.

---

[30] Reference from https://torstat.xenobite.eu

Tor network visualization website

To show the information of the background database of Tor network, we set up a website with all server nodes' information and provide some retrieve functions for the database.



Tor Node Information

Choose Search Type:

Nick Name

Enter Search Term:

Search

OR Enter Yourself SQL Command:

Example:

select * from tor_nodes where IsFast = "1" and IsGuard = "1"

Search

All of Tor node now: 2183

No	Nick Name	Descriptor Hash	Publication Date	Publication Time	IP	OR	DIR	FQDN	CC	Is Authority	Is Exit	Is Fast
1	Unnamed	PpbellUsfLjrpermGBAmPZkkE2w	2008-04-26	15:47:14	91.64.117.206	9001	9030	91-64-117-206-dynip.superkabel.de	DE	0	1	0
2	torAdm84sn001	DJb9h9vCC8APbzKC3wi2MM+HMCc	2008-04-26	16:59:55	194.102.180.118	9001	0	07.airg.sibiu.astral.ro	RO	0	1	1
3	bigbear3001	FYUEIrh52sEf3uT2taHRD97Arbk	2008-04-26	18:02:13	88.198.72.170	9001	9030	www.perhab.com	DE	0	1	1
4	AnnaStormone	5YxcaMqoXC7qgXJshlvZZMECZDM	2008-04-26	19:10:14	61.135.253.185	9080	443	61.135.253.185	CN	0	0	1
5	mx118	JtjB0m8/4ZDKStXqXESi9MdOWE0	2008-04-26	17:09:04	193.125.39.54	9001	0	host-193-125-39-54.real.kvidex.ru	RU	0	1	1
6	Unnamed	VLP0tdL61Hur2WFTkzKyYA6bPHU	2008-04-26	18:14:24	116.49.110.87	443	9030	n11649110087.netvigator.com	HK	0	0	0
7	Lucna	GpXcE3qGLi5aT4zHw+THAOEP38c	2008-04-29	02:34:22	216.110.217.24	9001	9030	216.110.217.24	US	0	0	0
8	kalle	9LaA7rhZOLr342g8aeNwmTizbw	2008-04-26	16:14:32	213.113.41.9	443	9030	c-092971d5.011-52-626c671.cust.bredbandsbolaget.se	SE	0	1	0
9	thorfintor	UkE4S+6g8H3M64wGiVZuchZMaCc	2008-04-29	02:27:39	89.110.146.219	9001	9030	z-at.de	DE	0	0	1
10	neunterNovember	URY5LZOoXTnMiZ1pBDybFzaPYzo	2008-04-26	21:25:19	91.13.255.215	16487	16488	p5b0dffd7.dip.t-dialin.net	DE	0	0	0
11	ForTheFreedom	I7esOEVuZ3aOY2D1GsElvizoXok	2008-04-29	05:50:17	96.234.81.136	443	9030	pool-96-234-81-136.nwrknj.fios.verizon.net	US	0	1	0

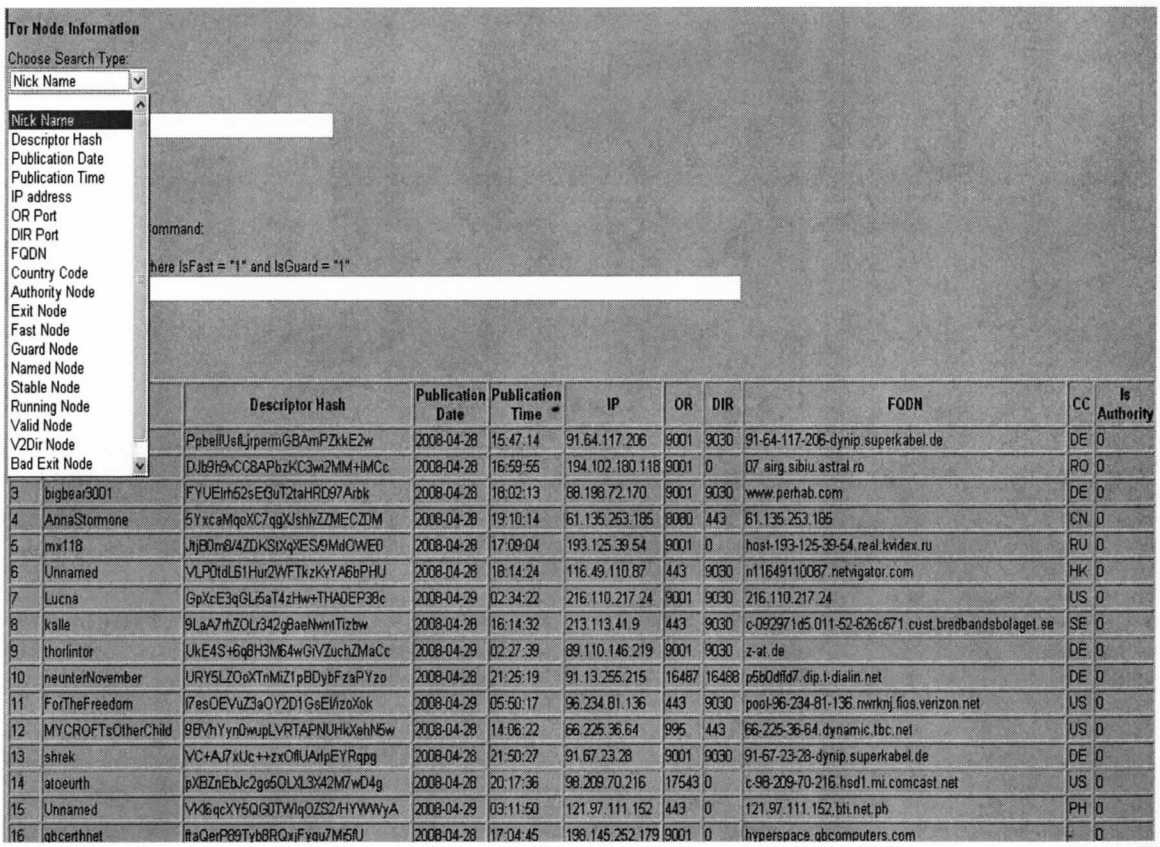
Figure 33 Tor network visualization website

The website shows comprehensive information of Tor server nodes such as Nick Name, IP address, Descriptor Hash, Country code, flags, and so on. The website uses PHP

script embedded in HTML code to connect and retrieve database remotely. Now the website address is http://66.214.163.6/torsearchindexv2.php .

The website is published with Apache web server application. The website page has two parts that are retrieve interface and the table for nodes' information. About the retrieve interface, there are two kinds of search interfaces. One is the key work search function, and the other is SQL statement search function that permit website visitors to enter SQL retrieve sentence to get the customized retrieve conditions.

For the first search function, key word search, as showed on the following figure, the website can use many key words to look up the nodes. For example we can use Country code, IP address, Nickname, and so on to search the qualified server nodes in the database. We will give some examples about this search function.



		Descriptor Hash	Publication Date	Publication Time	IP	OR	DIR	FQDN	CC	Is Authority
		PpbellUsfLjrpermGBAmPZkkE2w	2008-04-28	15:47:14	91.64.117.206	9001	9030	91-64-117-206-dynip.superkabel.de	DE	0
		DJb9h9vCC8APbzKC3wi2MM+lMCc	2008-04-28	16:59:55	194.102.180.118	9001	0	07.airg.sibiu.astral.ro	RO	0
3	bigbear3001	FYUElrh52sEf3uT2taHRD97Arbk	2008-04-28	18:02:13	88.198.72.170	9001	9030	www.perhab.com	DE	0
4	AnnaStormone	5YxcaMqoXC7qgXJshlvZZMECZDM	2008-04-28	19:10:14	61.135.253.185	8080	443	61.135.253.185	CN	0
5	mx118	JhjB0m8/4ZDKStXqXESi9MdOWE0	2008-04-28	17:09:04	193.125.39.54	9001	0	host-193-125-39-54.real.kvidex.ru	RU	0
6	Unnamed	VLP0tdL61Hur2WFTkzKvYA6bPHU	2008-04-28	18:14:24	116.49.110.87	443	9030	n116491100B7.netvigator.com	HK	0
7	Lucna	GpXcE3qGL5aT4zHw+THA0EP38c	2008-04-29	02:34:22	216.110.217.24	9001	9030	216.110.217.24	US	0
8	kalle	9LaA7rhZOLr342gBaeNwmTizbw	2008-04-28	16:14:32	213.113.41.9	443	9030	c-092971d5.011-52-626c671.cust.bredbandsbolaget.se	SE	0
9	thorlintor	UkE4S+6q8H3M64wGiVZuchZMaCc	2008-04-29	02:27:39	89.110.146.219	9001	9030	z-at.de	DE	0
10	neunterNovember	URY5LZOoXTnMiZ1pBDybFzaPYzo	2008-04-28	21:25:19	91.13.255.215	16487	16488	p5b0dffd7.dip.t-dialin.net	DE	0
11	ForTheFreedom	l7esOEVuZ3aOY2D1GsElrizoXok	2008-04-29	05:50:17	96.234.81.136	443	9030	pool-96-234-81-136.nwrknj.fios.verizon.net	US	0
12	MYCROFTsOtherChild	9BVhYyn0wupLVRTAPNUHkXehN5w	2008-04-28	14:06:22	66.225.36.64	995	443	66-225-36-64.dynamic.tbc.net	US	0
13	shrek	VC+AJ7xUc++zxOflUAxlpEYRqpg	2008-04-28	21:50:27	91.67.23.28	9001	9030	91-67-23-28-dynip.superkabel.de	DE	0
14	atoeurth	pXBZnEbJc2go5OLXL3X42M7wD4g	2008-04-28	20:17:36	98.209.70.216	17543	0	c-98-209-70-216.hsd1.mi.comcast.net	US	0
15	Unnamed	VKl6qcXY5QG0TWlqOZS2/HYWWyA	2008-04-29	03:11:50	121.97.111.152	443	0	121.97.111.152.bti.net.ph	PH	0
16	qbcerthnet	ftaQerP89TybBRQxjFyqu7Mi5fU	2008-04-28	17:04:45	198.145.252.179	9001	0	hyperspace.qbcomputers.com	-	0

Figure 34 Search criteria of the website

We select Nick Name as the key word to retrieve exact one node fitting the key word standard. Here we enter "bigbear3001" as the search word. We get the following search result satisfying our retrieve name.

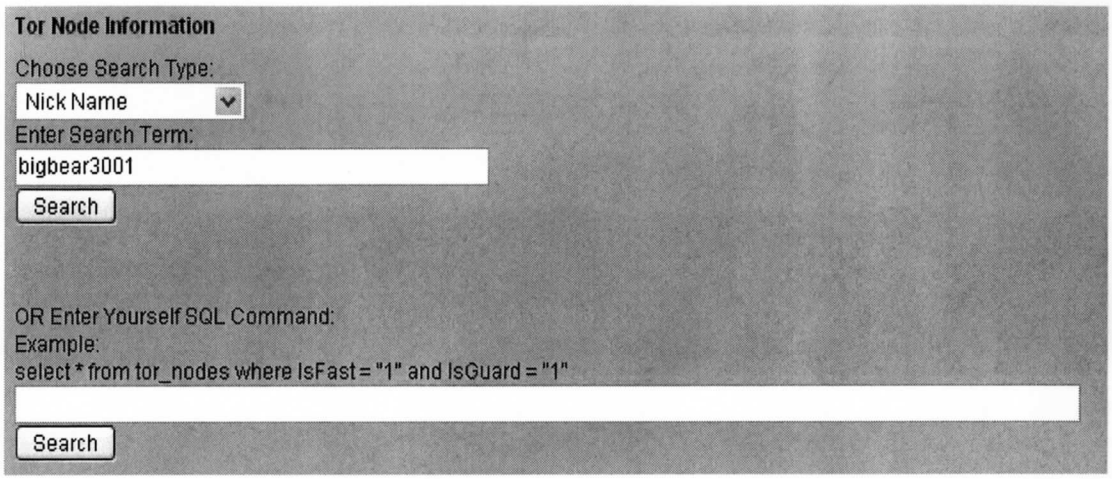


Figure 35 Use NickName as search keyword

Number of Tor node found: 1

No	Nick Name	Descriptor Hash	Publication Date	Publication Time	IP	OR	DIR	FQDN	CC	Is Authority	Is Exit	Is Fa
1	bigbear3001	FYUElrh52sEf3uT2taHRD97Arbk	2008-04-28	18:02:13	88.198.72.170	9001	9030	www.perhab.com	DE	0	1	1

Figure 36 The result of search with Nick Name condition

The node satisfying the retrieve condition has the same value for Nick Name field to the key word we entered just now.

We select Country Code as the retrieve condition to get the nodes from one country. E.g. we enter DE into the search key word and select Country Code as the search standard. We get the following retrieve result web page. All the values of Country Code field are "DE".

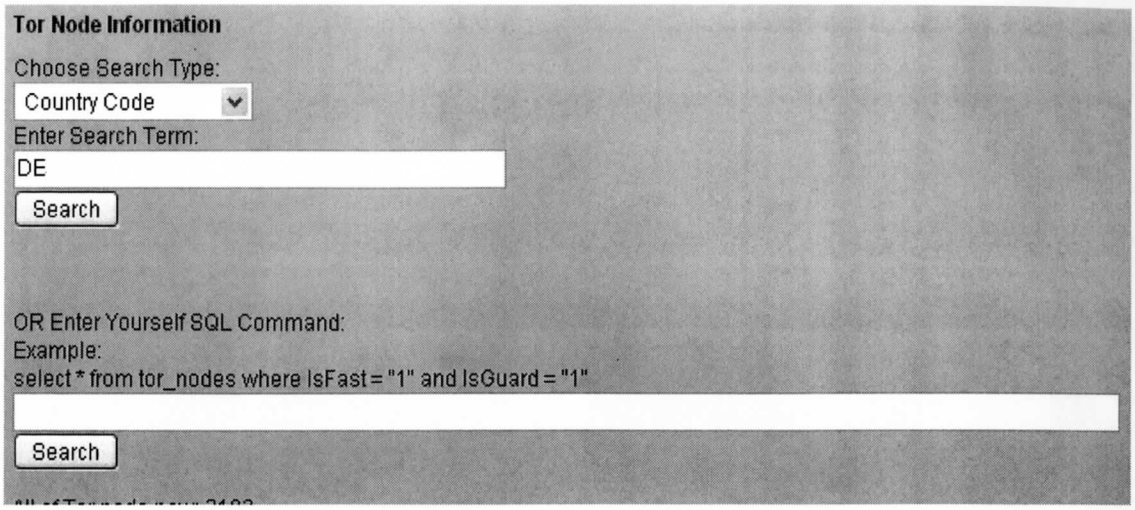


Figure 37 Use Country Code as search keyword

Number of Tor node found: 634

No	Nick Name	Descriptor Hash	Publication Date	Publication Time	IP	OR	DIR	FQDN	CC	I Auth
1	Unnamed	PpbeIIUsfLjrpermGBAmPZkkE2w	2008-04-28	15:47:14	91.64.117.206	9001	9030	91-64-117-206-dynip.superkabel.de	DE	0
2	bigbear3001	FYUElrh52sEf3uT2taHRD97Arbk	2008-04-28	18:02:13	88.198.72.170	9001	9030	www.perhab.com	DE	0
3	thorlintor	UkE4S+6q8H3M64wGiVZuchZMaCc	2008-04-29	02:27:39	89.110.146.219	9001	9030	z-at.de	DE	0
4	neunterNovember	URY5LZOoXTnMtZ1pBDybFzaPYzo	2008-04-28	21:25:19	91.13.255.215	16487	16488	p5b0dffd7.dip.t-dialin.net	DE	0
5	shrek	VC+AJ7zUc++zxOfIUArIpEYRqpg	2008-04-28	21:50:27	91.67.23.28	9001	9030	91-67-23-28-dynip.superkabel.de	DE	0
6	alliumcepa	n0aXf1P0vp27ZVBt7kBDae6KV4M	2008-04-29	03:01:53	84.63.103.83	9001	0	dslb-084-063-103-083.pools.arcor-ip.net	DE	0
7	henck2	oi6BPR4WLWFAwrGsDPcsb60xRsk	2008-04-28	15:04:06	81.169.168.60	9090	0	yog-sothoth.naggel.com	DE	0
8	LordBandicore	lwEugBdnr3EtCwTbUOYHoKzt12A	2008-04-28	15:23:27	213.196.206.35	9001	0	xdsl-213-196-206-35.netcologne.de	DE	0
9	Unnamed	crqL0BOq+Zd+7hHt6eKo0DM10hE	2008-04-28	16:38:38	77.178.153.175	9001	0	dtmd-4db299af.pool.einsundeins.de	DE	0
10	EntitaetNull	fIJ05yWPeLgq6pZdbiq7wFKEurY	2008-04-29	04:43:20	81.169.142.26	4242	4243	anonymizer.zesstra.de	DE	0
11	gray	SCizeWkPKfmKT196fgBouSFPHE	2008-04-28	18:48:38	213.239.220.75	9001	9030	gray.tabularazor.org	DE	0
12	glookleCDN	kkCpLGdvitdL4A4s+ZKWQGgX9J4	2008-04-28	14:01:02	89.149.242.226	9001	9030	glookle.com	DE	0
13	StasiServer	Iw2lXHet8hfPA2g58UubQD8MguA	2008-04-28	17:54:03	92.228.10.14	9001	9030	g228010014.adsl.alicedsl.de	DE	0
14	Gehr	/rLgylXMDtOjxTHI21JijXCq7EM	2008-04-29	06:37:18	85.177.183.131	9001	9030	e177183131.adsl.alicedsl.de	DE	0
15	papifartsfast	re9UMuWw1d+uK4+GML6ZYGsnpsM	2008-04-29	00:53:52	84.56.47.248	9001	0	dslb-084-056-047-248.pools.arcor-ip.net	DE	0
16	Unnamed	bzL11lUyOcnIrvNImfoTA90Rm0A	2008-04-29	05:52:24	78.94.84.220	9001	9030	ip-78-94-84-220.hsi.ish.de	DE	0
17	Snuffle	WJ6Ryy+DWgmMqnll4kgTQZTEeq4	2008-04-29	05:58:49	89.12.217.211	9001	0	manz-590cd9d3.pool.einsundeins.de	DE	0
18	autobahn	UWKF1EGcmCXidqtOZdMtAz3DcD4	2008-04-28	20:51:57	92.226.209.45	9001	0	g226209045.adsl.alicedsl.de	DE	0
19	testortwo	eklYrjT9O0UgOo8R0yzDMdamhqc	2008-04-28	13:40:48	88.84.144.63	9002	0	v29963.1blu.de	DE	0

Figure 38 Retrieve result for Country Code condition

We can select all flag statuses as search key word to retrieve the nodes with special flag status. The flag statues include "Is Authority, Is Exit, Is Fast, Is Guard, Is Named, Is Stable, Is Running, Is valid, Is V2Dir, Is BadExit, and Is BadDirectory".

Here we select one of the above flags as the search key word to make an example. We can get the following result web page with "Is Stable" search key word. As showed on the web page, we get 647 nodes with IsStable status.

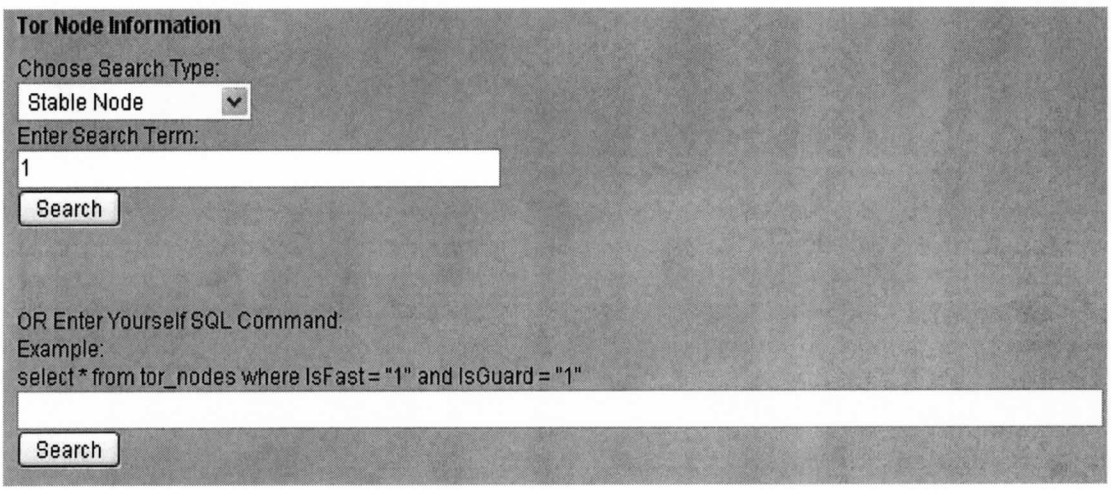


Figure 39 Use Stable Node as search keyword

Number of Tor node found: 647

No	Nick Name	Descriptor Hash	Publication Date	Publication Time	IP	OR	DIR	FQDN	CC	Is Authority	Is Exit	Is Fast	Is Guard	Is Named	Is Stable
1	bigbear3001	FYUEIth52sEf3uT2taHRD97Arbk	2008-04-28	18:02:13	88.198.72.170	9001	9030	www.perhsb.com	DE	0	1	1	1	1	1
2	AnnaStormone	5YxraMqoXC7qgXJshlvZZMECZ0M	2008-04-28	19:10:14	61.135.253.185	8080	443	61.135.253.185	CN	0	0	1	0	1	1
3	mx118	JtjB0m8/4ZDKStXqXESi9MdOWB0	2008-04-28	17:09:04	193.125.39.54	9001	0	host-193-125-39-54.real.kvidex.ru	RU	0	1	1	0	1	1
4	thorlintor	UkE4S+6q8H3M64wGiVZuchZMaCc	2008-04-29	02:27:39	89.110.146.219	9001	9030	z-at.de	DE	0	0	1	1	1	1
5	atoeurth	pXBZnEbJc2go5OLXL3X42M7wD4g	2008-04-28	20:17:36	98.209.70.216	17543	0	c-98-209-70-216.hsd1.mi.comcast.net	US	0	0	1	0	1	1
6	gbcerthnet	ftaQerP89Tyb8RQqFygu7Mi5fU	2008-04-28	17:04:45	198.145.252.179	9001	0	hyperspace.gbcomputers.com	-	0	0	1	0	1	1
7	NetworkPimp	+41qfc+NwJu2ta7Q+uKINWS6XA0	2008-04-28	17:17:23	216.27.177.188	443	0	ho.networkpimp.org	US	0	1	1	0	1	1
8	GravenResearch1	Vgf4plbGGItqVgbuXbNRBpE9rd	2008-04-28	21:09:43	60.242.100.67	9001	0	60-242-100-67.static.tpgi.com.au	AU	0	0	1	0	1	1
9	BinaryBLENDER	JwFcVj5baJNkfuq4bbfB01TfiZc	2008-04-28	22:14:08	67.100.94.85	443	0	host85.telizon.net	-	0	0	1	0	1	1
10	EntitaetNull	flJ05yWPeLgq6pZdbiq7wFKEurY	2008-04-29	04:43:20	81.169.142.26	4242	4243	anonymizer.zesstra.de	DE	0	0	1	1	1	1
11	gray	SCizeWkPKfmKT196fgBouSFPHE	2008-04-28	18:48:38	213.239.220.75	9001	9030	gray.tabularazor.org	DE	0	0	1	1	1	1
12	salottisipuli	2HhvO62XVEb0iJTzpBDfN9YnwhY	2008-04-28	21:28:10	91.152.67.26	53280	0	a91-152-67-26.elisa-laajakaista.fi	FI	0	0	1	0	1	1
13	whomthebelltolls	XCKUYpIvSgG885/tqkMUQS1t0zE	2008-04-28	17:09:38	72.43.122.208	9001	9030	rrcs-72-43-122-208.nyc.biz.rr.com	-	0	1	1	1	0	1
14	nixnix	rS7BiiEpvMF2+ToHWnNw0U1mW6M	2008-04-29	05:50:01	149.9.0.58	9001	9030	149.9.0.58	US	0	0	1	1	0	1
15	67YhhtG565543TGRf	R9DPm2S723SfDzXYEwp/WyhPBvA	2008-04-28	14:02:47	202.60.234.34	9001	0	202.60.234.034.static.cyberec.com	HK	0	0	1	0	1	1
16	bananasana	NJxJkEIhfSSOFAF1U27BY4/A6+M	2008-04-28	18:02:07	195.20.207.162	9001	0	last.endnode.se	SE	0	0	1	0	1	1
17	host112110	pXBAwKYeYk0mMJ4jOQdfGWoKpZI	2008-04-29	06:38:39	200.53.120.110	9001	9030	host112110.metrored.net.mx	MX	0	0	1	1	1	1
18	devsecurerandom	OqTVyAHxp7gQNyr6Vki+1ccPWFI	2008-04-29	03:01:17	80.74.155.121	9001	0	vz-rooney.sui-inter.net	CH	0	0	1	0	1	1
19	Yo4	r4pbIWHP2m6oyiudi6sV5CkVo24s	2008-04-28	16:21:24	168.150.251.13	9091	9092	dcn251-13.dcn.davis.ca.us	US	0	0	1	1	1	1
20	pickaproxy	ZNl6CF3Cy4AT/8fjoHD8o8XIcY	2008-04-28	23:32:42	70.86.138.210	9001	9030	pickaproxy.com	-	0	0	1	1	1	1
21	MopperSmurf	kVNvc7+8CG4RJTjF8W1HJn92Pzc	2008-04-28	21:20:34	192.150.94.83	9001	9030	dhcp.as1101.net	NL	0	0	1	0	1	1
22	SEC	0dErrQavv0jBzXu0uUASK9sC0J4	2008-04-29	05:09:13	192.42.113.248	9001	9030	anonymous.sec.nl	NL	0	0	1	1	1	1

Figure 40 Retrieve result for IsStable condition

We can use IP address to search some nodes with special IP address. We enter one IP address to search one node with that IP address.

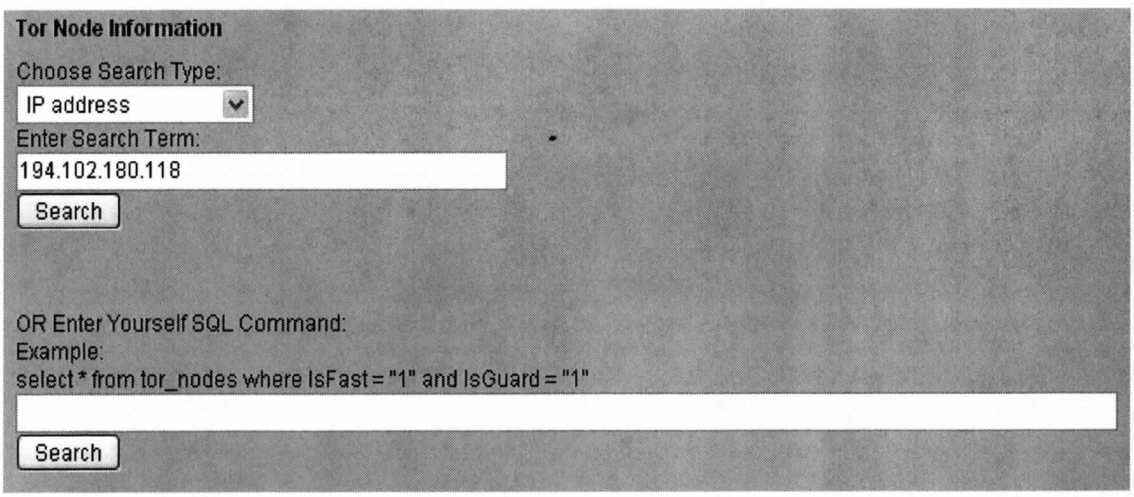


Figure 41 Use IP address as search keyword

Number of Tor node found: 1

No	Nick Name	Descriptor Hash	Publication Date	Publication Time	IP	OR	DIR	FQDN	CC	Is Authority	Is Exit
1	torAdm84srv001	DJb9h9vCC8APbzKC3wi2MM+IMCc	2008-04-28	16:59:55	194.102.180.118	9001	0	07.airg.sibiu.astral.ro	RO	0	1

Figure 42 Retrieve result for IP address condition

There is a second search type, writing SQL statement by visitor to retrieve the nodes satisfying the condition. All nodes' information are stored in a table named "tor_nodes". This way can let some visitors use their own SQL search condition to get the result. The following is the web page of this search way.

OR Enter Yourself SQL Command:
Example:
select * from tor_nodes where IsFast = "1" and IsGuard = "1"

Search

All of Tor node now: 2183

No	Nick Name	Descriptor Hash	Publication Date	Publication Time	IP	OR	DIR
1	Unnamed	PpbellUsfLjrpermGBAmPZkkE2w	2008-04-28	15:47:14	91.64.117.206	9001	9030
2	torAdm84srv001	DJb9h9vCC8APbzKC3wi2MM+IMCc	2008-04-28	16:59:55	194.102.180.118	9001	0
3	bigbear3001	FYUElrh52sEf3uT2taHRD97Arbk	2008-04-28	18:02:13	88.198.72.170	9001	9030
4	AnnaStormone	5YxcaMqoXC7qgXJshlvZZMECZ0M	2008-04-28	19:10:14	61.135.253.185	8080	443

Figure 43 SQL sentence search type

We will give one "AND" and one "OR" example to see this search function. For example, we enter one SQL sentence that searches the nodes with combining condition of IsFast and IsGuard. The SQL sentence is as follows: ' select * from tor_nodes where IsFast = "1" and IsGuard = "1" '. There are 344 nodes with this SQL sentence condition.

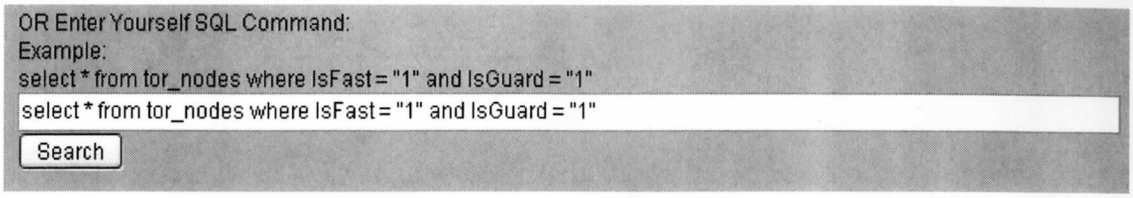


Figure 44 Enter one SQL "AND" sentence as search way

Number of Tor node found: 344

No	Nick Name	Descriptor Hash	Publication Date	Publication Time	IP	OR	DIR	FQDN	CC	Is Authority	Is Exit	Is Fast	Is Guard
1	bigbear3001	FYUEIrh52sEf3uT2taHRD97Arbk	2008-04-28	18:02:13	88.198.72.170	9001	9030	www.perhab.com	DE	0	1	1	1
2	thorlintor	UkE4S+6q8H3M64wGiVZuchZMaCc	2008-04-29	02:27:39	89.110.146.219	9001	9030	z-at.de	DE	0	0	1	1
3	shrek	VC+AJ7zUc++zxOfIUArIpEYRqpg	2008-04-28	21:50:27	91.67.23.28	9001	9030	91-67-23-28-dynip.superkabel.de	DE	0	0	1	1
4	EntitaetNull	flJ05yWPeLgq6pZdbiq7wFKEurY	2008-04-29	04:43:20	81.169.142.26	4242	4243	anonymizer.zesstra.de	DE	0	0	1	1
5	grey	SCizeWkPKfmKT196fgiBouSFPHE	2008-04-28	18:48:38	213.239.220.75	9001	9030	grey.tabularazor.org	DE	0	0	1	1
6	whomthebelltolls	XCKUYplvSgG885/lqkMUQS1t0zE	2008-04-28	17:09:38	72.43.122.208	9001	9030	rrcs-72-43-122-208.nyc.biz.rr.com	-	0	1	1	1
7	nixnix	rS7Bi8EpvMF2+ToHWnNw0U1mW6M	2008-04-29	05:50:01	149.9.0.58	9001	9030	149.9.0.58	US	0	0	1	1
8	host112110	pXBAwKYeYk0mMJ4jOQdfGWoKpZI	2008-04-29	06:38:39	200.53.120.110	9001	9030	host112110.metrored.net.mx	MX	0	0	1	1
9	Yo4	r4pbIWHP2m6oyiud6sV5CkVo24s	2008-04-28	16:21:24	168.150.251.13	9091	9092	dcn251-13.dcn.davis.ca.us	US	0	0	1	1
10	pickaproxy	ZNI6CF3Cy4AT/8fjoHD8o/8XIcY	2008-04-28	23:32:42	70.86.138.210	9001	9030	pickaproxy.com	-	0	0	1	1
11	SEC	0dErzQavv0jBzXs0uUASK9sC0J4	2008-04-29	05:09:13	192.42.113.248	9001	9030	anonymous.sec.nl	NL	0	0	1	1
12	hamakor	Vs7f/8i/9y+T8cEkQiOQa7QXRCc	2008-04-28	15:12:09	82.80.248.177	443	80	bzq-82-80-248-177.dcenter.bezeqint.net	IL	0	0	1	1
13	TaurNuFuin	gpK03jKoCorgg0KFGneZP+j6rCM	2008-04-29	01:27:43	77.190.91.175	9001	9030	essn-4dbe5baf.pool.einsundeins.de	DE	0	0	1	1
14	bloxortsipt38	xFyswdocWNQjm82TW0MoaYlMN0K8	2008-04-29	05:53:29	209.169.100.45	9001	9030	conr-adsl-209-169-100-45.consolidated.net	US	0	0	1	1
15	SuOldec9	qchpMWM19bN0iFzMgMvBh/QoRW8	2008-04-29	00:05:56	90.49.20.99	9001	9030	anantes-158-1-125-99.w90-49.abo.wanadoo.fr	FR	0	0	1	1
16	codemonkeysorg	DR+FKAJZMw3S0VOqlTgnLTBtMm8	2008-04-29	04:49:08	38.119.53.61	9001	9030	tor.code-monkeys.org	-	0	0	1	1
17	ZesstrasLair	gwd8j8q9wl/qOueFQg8FmXp8O9c	2008-04-28	16:20:02	85.214.85.116	4242	0	h1394605.stratoserver.net	DE	0	0	1	1
18	24E6F91D5016	OWmSlIRgAphOOODDopjbnRCr6k	2008-04-29	02:35:49	67.218.193.90	9001	0	delta.comeau.info	-	0	0	1	1
19	daedalus	MY5jeEMJCbYRoO0LgXzEUER+Okl	2008-04-29	05:46:59	84.157.160.61	9001	9030	p549da03d.dip0.t-ipconnect.de	DE	0	0	1	1
20	robotical337	P+D2oK33xyzsNFLxo5vE25d/51E	2008-04-28	17:51:13	87.120.199.10	9991	0	mail.p2p-bg.net	BG	0	1	1	1
21	NSAFortMeade	Sbhu/TGzpYeH36rhfqtk3NHPC9k	2008-04-28	21:19:57	206.222.29.14	4443	8080	tor.epipe.com	-	0	0	1	1
22	snIP3r2	0gFbwFUpYIJle1EFF9N9Ouw6heo	2008-04-28	22:04:07	85.214.54.254	9001	0	it-wagner.net	DE	0	0	1	1
23	1000rpmLinux	0+yd8t4eMzUC+xDhm/qK4+ofjc	2008-04-28	15:11:26	213.114.108.141	9001	9030	c-8d6c72d5.017-62-6b73642.cust.bredbandsbolaget.se	SE	0	1	1	1

Figure 45 result for SQL "AND" sentence

We also can enter "OR" SQL sentence to search nodes satisfying one or all conditions. We enter 'select * from tor_nodes where IsFast = "1" or IsGuard = "1" ' into the search field. We get 1181 nodes with one or both condition of IsFast and IsGuard.

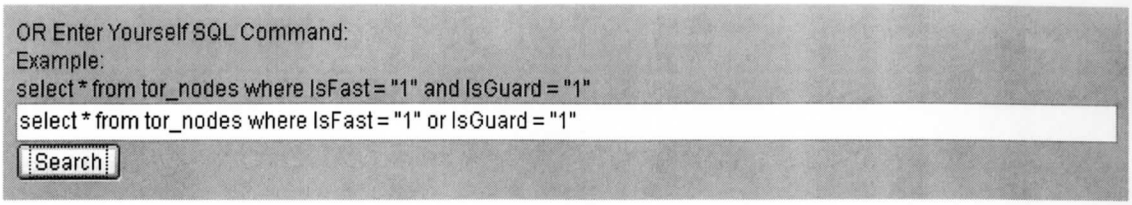


Figure 46 Enter one SQL "OR" sentence as search way

Number of Tor node found: 1181

No	Nick Name	Descriptor Hash	Publication Date	Publication Time	IP	OR	DIR	FQDN	CC	Is Authority	Is Exit	Is Fast	Is Guard
1	torAdm84srv001	DJb9h9vCC8APbzKC3wi2MM+HMCc	2008-04-28	16:59:55	194.102.180.118	9001	0	07.airg.sibiu.astral.ro	RO	0	1	1	0
2	bigbear3001	FYUEIth52sEf3uT2taHRD97Arbk	2008-04-28	18:02:13	88.198.72.170	9001	9030	www.perhab.com	DE	0	1	1	1
3	AnneStormone	5YxcaMqoXC7qgXJshlvZZMECZ0M	2008-04-28	19:10:14	61.135.253.185	8080	443	61.135.253.185	CN	0	0	1	0
4	mx118	JtjB0m8/4ZDKStXqXES/9MdOWE0	2008-04-28	17:09:04	193.125.39.54	9001	0	host-193-125-39-54.real.kvidex.ru	RU	0	1	1	0
5	thorlintor	UkE4S+6q8H3M64wGiVZuchZMaCc	2008-04-29	02:27:39	89.110.146.219	9001	9030	z-at.de	DE	0	0	1	1
6	MYCROFTsOtherChild	9BVhYyn0wupLVRTAPNUHkXehN5w	2008-04-28	14:06:22	66.225.36.64	995	443	66-225-36-64.dynamic.tbc.net	US	0	1	1	0
7	shrek	VC+AJ7xUc++zxOfIUArIpEYRqpg	2008-04-28	21:50:27	91.67.23.28	9001	9030	91-67-23-28-dynip.superkabel.de	DE	0	0	1	1
8	atoeurth	pXBZnEbJc2go3OLXL3X42M7wD4g	2008-04-28	20:17:36	98.209.70.216	17543	0	c-98-209-70-216.hsd1.mi.comcast.net	US	0	0	1	0
9	gbcerthnet	ftaQerP89Tyb8RQxjFygu7Mi5fU	2008-04-28	17:04:45	198.145.252.179	9001	0	hyperspace.gbcomputers.com	-	0	0	1	0
10	NetworkPimp	+41qfc+NwJu2ta7Q+uKlNWS6XA0	2008-04-28	17:17:23	216.27.177.188	443	0	ho.networkpimp.org	US	0	1	1	0
11	alliumcepa	n0aXf1P0vp27ZVBt7kEDae6KV4M	2008-04-29	03:01:53	84.63.103.83	9001	0	dslb-084-063-103-083.pools.arcor-ip.net	DE	0	1	1	0
12	userPC	cDUXLz78YSTh5R72SFCc1lf0o	2008-04-29	06:29:06	87.205.184.76	443	9030	87-205-184-76.adsl.inetia.pl	PL	0	1	1	0
13	GravenResearch1	Vgf4plbGHtqVgbuXbNRBpE/9zI	2008-04-28	21:09:43	60.242.100.67	9001	0	60-242-100-67.static.tpgi.com.au	AU	0	0	1	0
14	hitler	eEyq+wkmgFT1mMypQxQQInc5/cs	2008-04-28	22:37:41	216.161.85.185	9001	0	tor.omgwallhack.org	US	0	1	1	0
15	BinaryBLENDER	JwFcVj5baJNkfuq4bbfB01TfiZc	2008-04-28	22:14:08	67.100.94.85	443	0	host85.telizon.net	-	0	0	1	0
16	EntitaetNull	fIJ05yWPeLgqópZdbiq7wFKEurY	2008-04-29	04:43:20	81.169.142.26	4242	4243	anonymizer.zesstra.de	DE	0	0	1	1
17	TheOnoinRules	zyGEIhJBvvkgTE3ecPR2ZhVS2bo	2008-04-28	17:47:32	216.171.141.126	443	0	transedge-141-126.transedge.com	US	0	1	1	0
18	gray	SCizeWkPKfmKT196fgiBouSFPHE	2008-04-28	18:48:38	213.239.220.75	9001	9030	gray.tabularazor.org	DE	0	0	1	1
19	salottisipuli	2HhvO62XVEb0JTzpBDfN9YnwhY	2008-04-28	21:28:10	91.152.67.26	53280	0	a91-152-67-26.elisa-laajakaista.fi	FI	0	0	1	0
20	whomthebelltolls	XCKUYpIvSgG885/lqkMUQS1t0xE	2008-04-28	17:09:38	72.43.122.208	9001	9030	rrcs-72-43-122-208.nyc.biz.rr.com	-	0	1	1	1
21	Unnamed	ZLTXDpcRFpwwhj24BC/3o/nVvHA	2008-04-29	06:04:45	119.32.25.166	9001	9030	119.32.25.166	CN	0	1	1	0
22	nixnix	rS7Bi8EpvMF2+ToHWnNw0U1mW6M	2008-04-29	05:50:01	149.9.0.58	9001	9030	149.9.0.58	US	0	0	1	1
23	glookleCDN	kkCpLGdvitdL4A4s+ZKWQGgX9J4	2008-04-28	14:01:02	89.149.242.226	9001	9030	glookle.com	DE	0	1	1	0

Figure 47 result for SQL "OR" sentence

With these two search ways, the website visitors can retrieve all nodes' information easily. It will improve the use of Tor network.

CHAPTER 4

CONCLUSIONS

After the planed and designed the database and website, I published the website with Apache web service application. The website link is http://66.214.163.6/torsearchindexv2.php . Through the website, we can check the detailed information for every Tor server nodes with the support of background database. The website and database can provide some information for Tor network users. It is good to promote the use of Tor network.

We also tested and analyzed Tor network and found out that the low performance of it is because the selection of low performance Tor server nodes. Finding out this weakness can help Tor developer change the software codes reasonably for the future version of Tor network.

In general, my project has gained the original objectives and has finished the planned scope.

REFERENCES

1. Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The Second-Generation Onion Router. Retrieved June 25, 2008, from http://www.torproject.org/svn/trunk/doc/design-paper/tor-design.html

2. Tor project organization. Tor knowledge and information. Retrieved June 26, 2008, from https://www.torproject.org/

3. Wikepedia. Tor (anonymity network). Retrieved June 27, 2008, from http://en.wikipedia.org/wiki/Tor_(anonymity_network)

4. Nadir Soualem. Crontab: Schedule Tasks. Retrieved June 27, 2008, from http://www.math-linux.com/spip.php?article45

5. Wikepedia. Traffic analysis. Retrieved June 27, 2008, from http://en.wikipedia.org/wiki/Traffic_analysis

6. Wikepedia. Symmetric-key algorithm, Retrieved June 27, 2008, from http://en.wikipedia.org/wiki/Symmetric_key

7. Wikepedia. Onion routing, Retrieved June 28, 2008, from http://en.wikipedia.org/wiki/Onion_routing

8. Wikepedia. Vidalia project, Retrieved June 29, 2008, from http://en.wikipedia.org/wiki/Vidalia_project

9. Wikepedia. Privoxy, Retrieved June 30, 2008, from http://en.wikipedia.org/wiki/Privoxy

10. Tor project organization. Tor Protocol Specification, Retrieved June 27, 2008, from https://www.torproject.org/svn/trunk/doc/spec/tor-spec.txt

APPENDICES

APPENDIX A: PROJECT MANAGEMENT DOCUMENTATION

Work Breakdown Structure (WBS)

1.0 Initiate Project Idea
- 1.1 Project Topic choosing
- 1.2 Prepare project idea
- 1.3 Research and Literature Review of project idea
- 1.4 Project timeline plan
- 1.5 Approval of project idea

2.0 Project Planning and Design
- 2.1 Project plan kickoff
- 2.2 Project objectives plan
  - 2.2.1 The objective of the reason for Tor low performance
  - 2.2.2 The objective of Tor information website
  - 2.2.3 The objective of Tor nodes database
- 2.3 Project scope plan
  - 2.3.1 The scope of the reason for Tor low performance
  - 2.3.2 The scope of Tor information website
  - 2.3.3 The scope of Tor nodes database
- 2.4 Project Work Breakdown Structure and Gantt Chart plan
  - 2.4.1 Review project objectives and scope

2.4.2 Create WBS

2.4.3 Schedule project tasks

2.4.4 Create Gantt Chart

2.4.5 Project deliverable plan

2.5 Set up the project committee

2.6 Approval of planning and design

3.0 Project Implementation

3.1 Project implementation kickoff

3.2 Find out the reason for Tor low performance

3.2.1 Set up the Tor test system

3.2.2 Test the existing Tor network

3.2.3 Analyze the test result

3.2.4 Find out the result of Tor low performance

3.3 Tor network nodes information website

3.3.1 Decide the content of the information website

3.3.2 Design the structure of the information website

3.3.3 Program the website

3.3.4 Test the information website

3.3.5 Tor information website completed

3.4 Tor network nodes database

3.4.1 Decide the content of the Tor database

3.4.2 Design the structure of the Tor database

3.4.3 Program Tor database

3.4.4 Link Tor database and Tor information website

3.4.5 Test Tor database

3.4.6 Tor database completed

3.5 Report to project committee about progress of implementation

4.0 Project Closing

4.1 Project presentation

4.2 Finalize paper work for project

4.2.1 write thesis

4.2.2 revise thesis

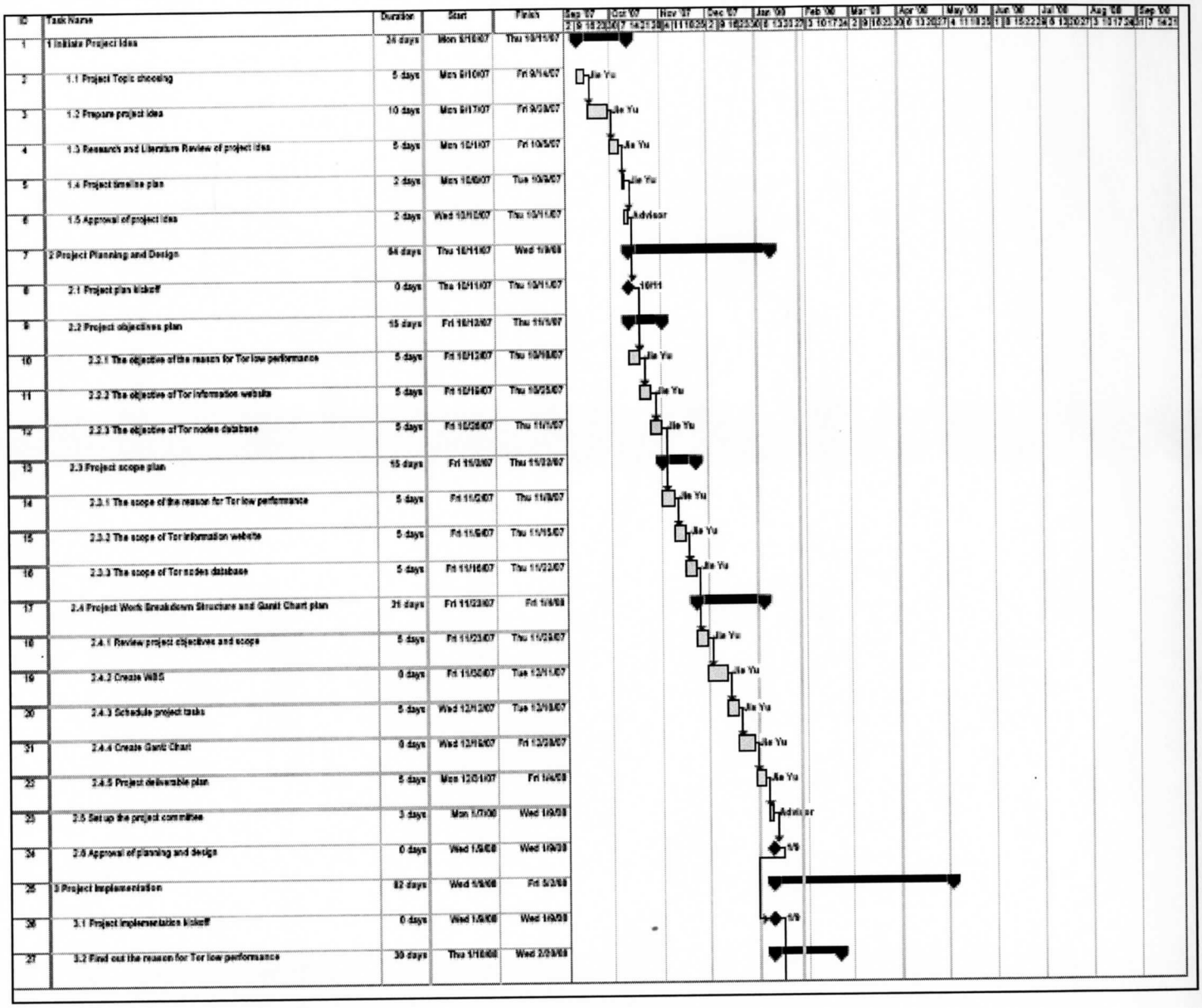
4.3 Report finalized paper work to project committee

4.4 Review final paper work of project by committee

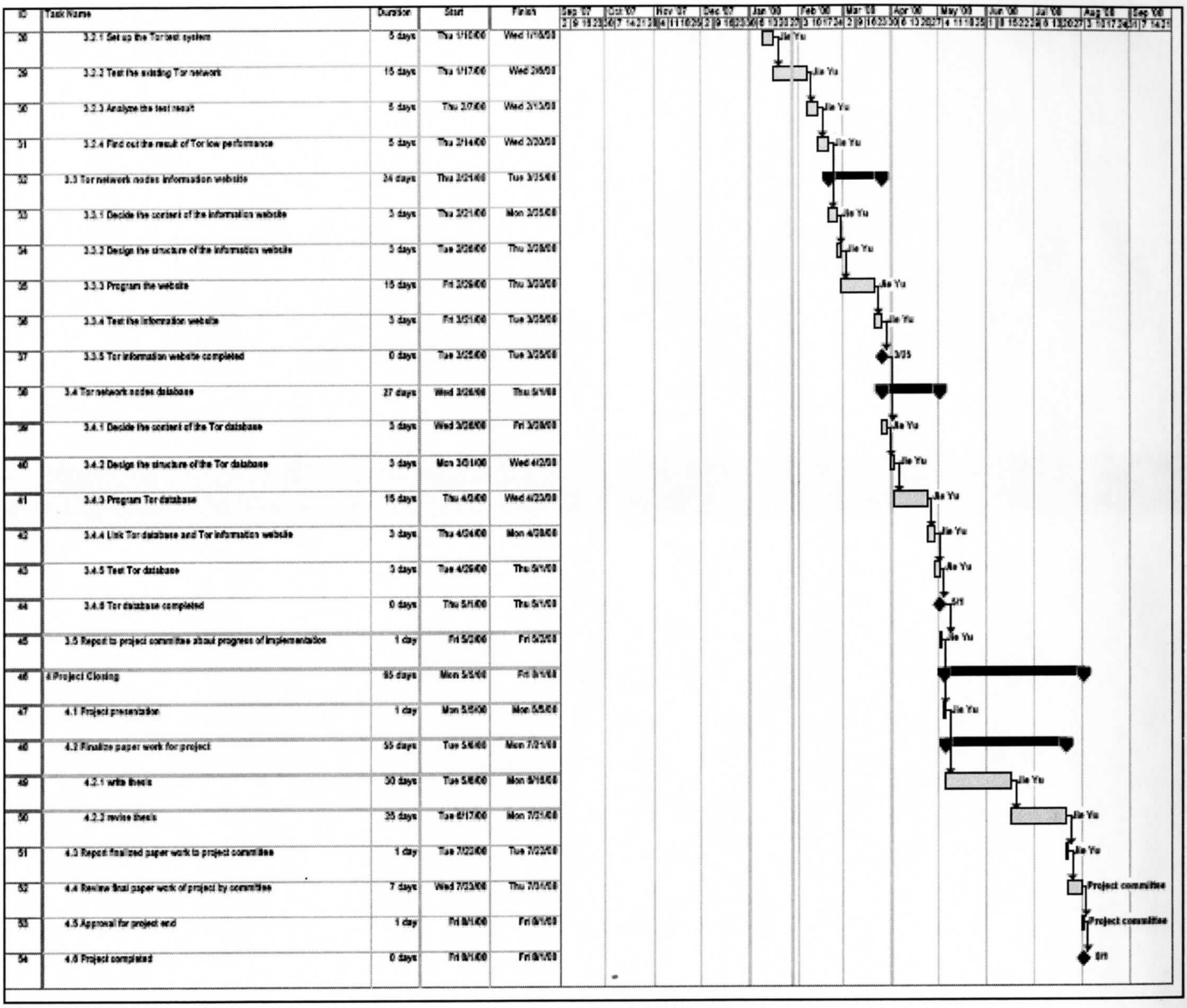
4.5 Approval for project end

4.6 Project completed

Gantt Chart



ID	Task Name	Duration	Start	Finish
1	1 Initiate Project Idea	24 days	Mon 9/10/07	Thu 10/11/07
2	1.1 Project Topic choosing	5 days	Mon 9/10/07	Fri 9/14/07
3	1.2 Prepare project idea	10 days	Mon 9/17/07	Fri 9/28/07
4	1.3 Research and Literature Review of project idea	5 days	Mon 10/1/07	Fri 10/5/07
5	1.4 Project timeline plan	2 days	Mon 10/8/07	Tue 10/9/07
6	1.5 Approval of project idea	2 days	Wed 10/10/07	Thu 10/11/07
7	2 Project Planning and Design	84 days	Thu 10/11/07	Wed 1/9/08
8	2.1 Project plan kickoff	0 days	Thu 10/11/07	Thu 10/11/07
9	2.2 Project objectives plan	15 days	Fri 10/12/07	Thu 11/1/07
10	2.2.1 The objective of the reason for Tor low performance	5 days	Fri 10/12/07	Thu 10/18/07
11	2.2.2 The objective of Tor information website	5 days	Fri 10/19/07	Thu 10/25/07
12	2.2.3 The objective of Tor nodes database	5 days	Fri 10/26/07	Thu 11/1/07
13	2.3 Project scope plan	15 days	Fri 11/2/07	Thu 11/22/07
14	2.3.1 The scope of the reason for Tor low performance	5 days	Fri 11/2/07	Thu 11/8/07
15	2.3.2 The scope of Tor information website	5 days	Fri 11/9/07	Thu 11/15/07
16	2.3.3 The scope of Tor nodes database	5 days	Fri 11/16/07	Thu 11/22/07
17	2.4 Project Work Breakdown Structure and Gantt Chart plan	31 days	Fri 11/23/07	Fri 1/4/08
18	2.4.1 Review project objectives and scope	5 days	Fri 11/23/07	Thu 11/29/07
19	2.4.2 Create WBS	0 days	Fri 11/30/07	Tue 12/11/07
20	2.4.3 Schedule project tasks	5 days	Wed 12/12/07	Tue 12/18/07
21	2.4.4 Create Gantt Chart	0 days	Wed 12/19/07	Fri 12/28/07
22	2.4.5 Project deliverable plan	5 days	Mon 12/31/07	Fri 1/4/08
23	2.5 Set up the project committee	3 days	Mon 1/7/08	Wed 1/9/08
24	2.6 Approval of planning and design	0 days	Wed 1/9/08	Wed 1/9/08
25	3 Project Implementation	82 days	Wed 1/9/08	Fri 5/2/08
26	3.1 Project implementation kickoff	0 days	Wed 1/9/08	Wed 1/9/08
27	3.2 Find out the reason for Tor low performance	30 days	Thu 1/10/08	Wed 2/20/08


ID
Task Name
Duration
Start
Finish
3.2.1 Set up the Tor test system
5 days
Thu 1/10/08
Wed 1/16/08
3.2.2 Test the existing Tor network
15 days
Thu 1/17/08
Wed 2/6/08
3.2.3 Analyze the test result
5 days
Thu 2/7/08
Wed 2/13/08
3.2.4 Find out the result of Tor low performance
5 days
Thu 2/14/08
Wed 2/20/08
3.3 Tor network nodes information website
24 days
Thu 2/21/08
Tue 3/25/08
3.3.1 Decide the content of the information website
3 days
Thu 2/21/08
Mon 2/25/08
3.3.2 Design the structure of the information website
3 days
Tue 2/26/08
Thu 2/28/08
3.3.3 Program the website
15 days
Fri 2/29/08
Thu 3/20/08
3.3.4 Test the information website
3 days
Fri 3/21/08
Tue 3/25/08
3.3.5 Tor information website completed
0 days
Tue 3/25/08
Tue 3/25/08
3.4 Tor network nodes database
27 days
Wed 3/26/08
Thu 5/1/08
3.4.1 Decide the content of the Tor database
3 days
Wed 3/26/08
Fri 3/28/08
3.4.2 Design the structure of the Tor database
3 days
Mon 3/31/08
Wed 4/2/08
3.4.3 Program Tor database
15 days
Thu 4/3/08
Wed 4/23/08
3.4.4 Link Tor database and Tor information website
3 days
Thu 4/24/08
Mon 4/28/08
3.4.5 Test Tor database
3 days
Tue 4/29/08
Thu 5/1/08
3.4.6 Tor database completed
0 days
Thu 5/1/08
Thu 5/1/08
3.5 Report to project committee about progress of implementation
1 day
Fri 5/2/08
Fri 5/2/08
4 Project Closing
85 days
Mon 5/5/08
Fri 8/1/08
4.1 Project presentation
1 day
Mon 5/5/08
Mon 5/5/08
4.2 Finalize paper work for project
55 days
Tue 5/6/08
Mon 7/21/08
4.2.1 write thesis
50 days
Tue 5/6/08
Mon 7/14/08
4.2.2 revise thesis
25 days
Tue 6/17/08
Mon 7/21/08
4.3 Report finalized paper work to project committee
1 day
Tue 7/22/08
Tue 7/22/08
4.4 Review final paper work of project by committee
7 days
Wed 7/23/08
Thu 7/31/08
4.5 Approval for project end
1 day
Fri 8/1/08
Fri 8/1/08
4.6 Project completed
0 days
Fri 8/1/08
Fri 8/1/08
Sep '07
Oct '07
Nov '07
Dec '07
Jan '08
Feb '08
Mar '08
Apr '08
May '08
Jun '08
Jul '08
Aug '08
Sep '08
Jie Yu
3/25
5/1
Project committee
8/1